Fig. 1

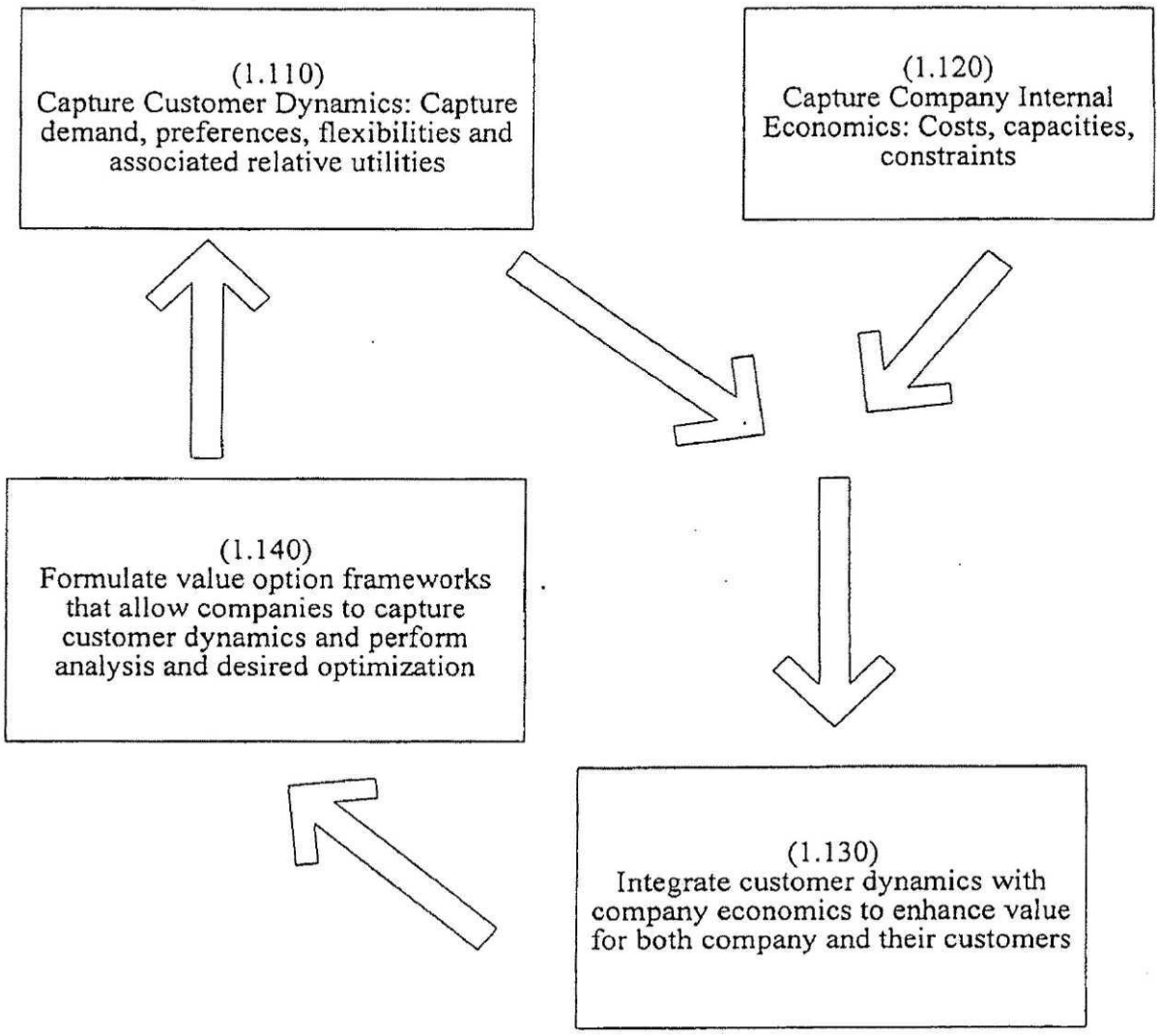

(1.110)
Capture Customer Dynamics: Capture demand, preferences, flexibilities and associated relative utilities
(1.120)
Capture Company Internal Economics: Costs, capacities, constraints
(1.140)
Formulate value option frameworks that allow companies to capture customer dynamics and perform analysis and desired optimization
(1.130)
Integrate customer dynamics with company economics to enhance value for both company and their customers

Fig. 2

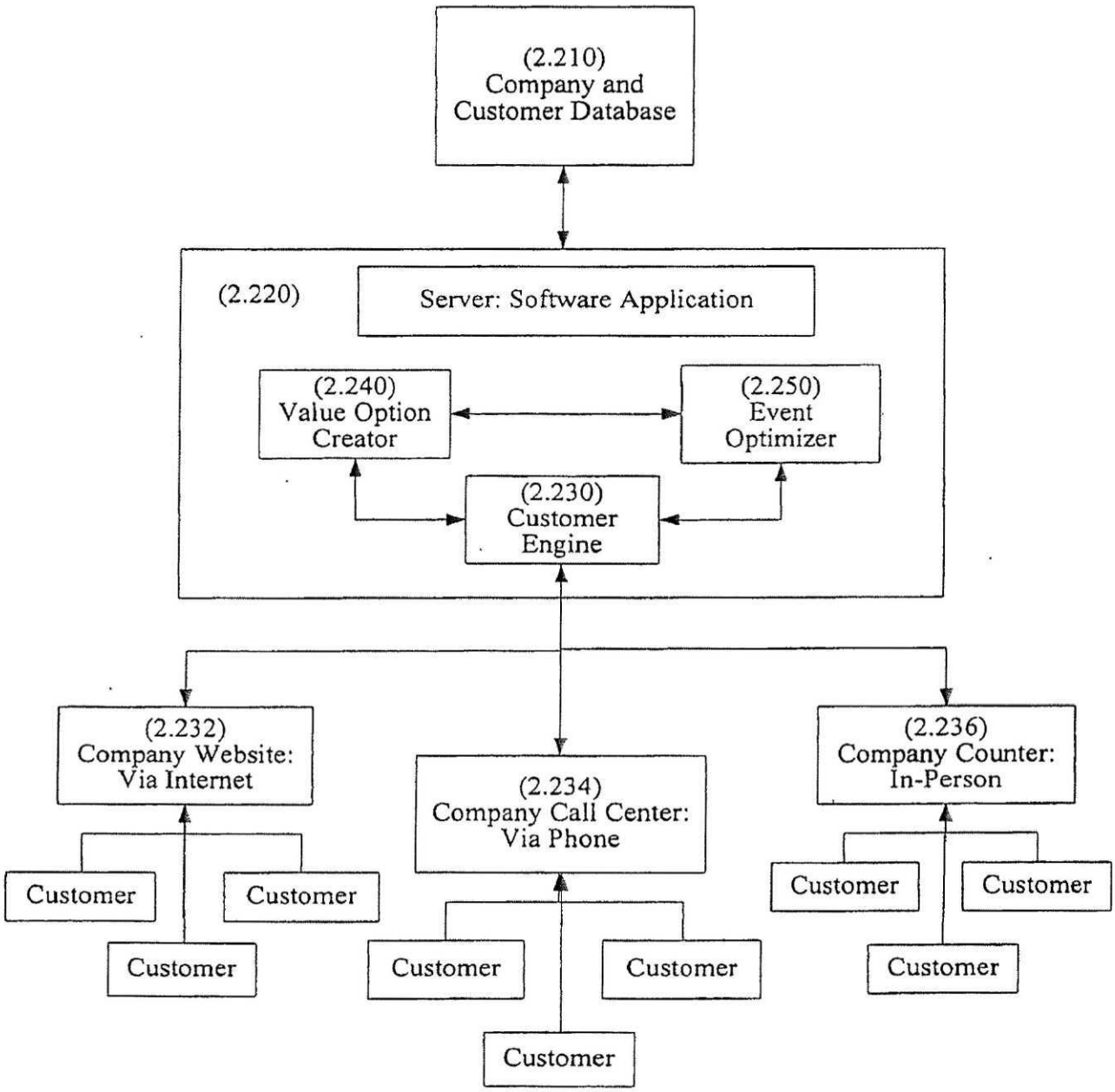

(2.210)
Company and
Customer Database
(2.220)
Server: Software Application
(2.240)
Value Option
Creator
(2.250)
Event
Optimizer
(2.230)
Customer
Engine
(2.232)
Company Website:
Via Internet
(2.234)
Company Call Center:
Via Phone
(2.236)
Company Counter:
In-Person
Customer
Customer
Customer
Customer
Customer
Customer
Customer
Customer
Customer

Fig. 3

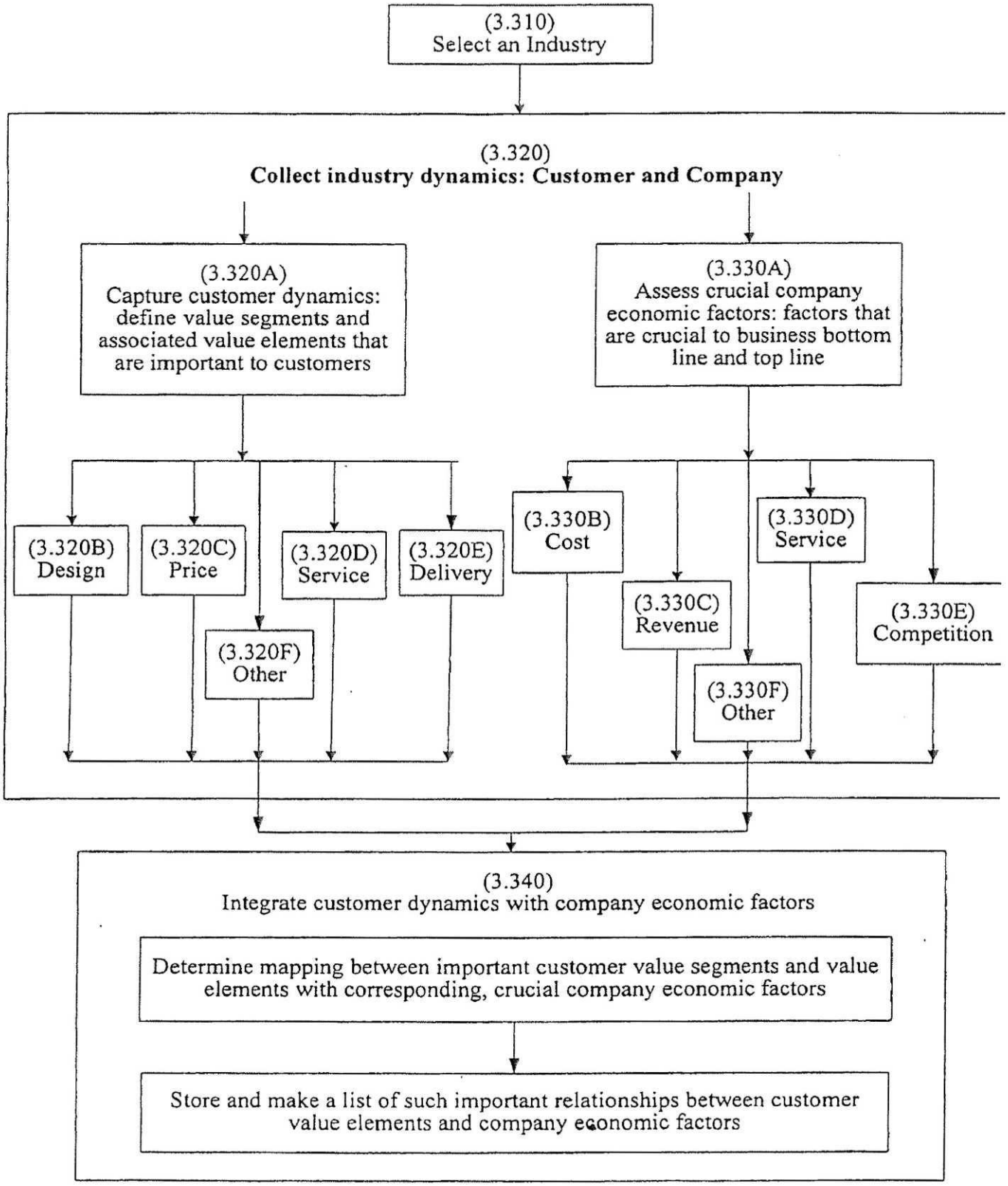

(3.310)
Select an Industry
(3.320)
Collect industry dynamics: Customer and Company
(3.320A)
Capture customer dynamics: define value segments and associated value elements that are important to customers
(3.330A)
Assess crucial company economic factors: factors that are crucial to business bottom line and top line
(3.320B)
Design
(3.320C)
Price
(3.320D)
Service
(3.320E)
Delivery
(3.320F)
Other
(3.330B)
Cost
(3.330C)
Revenue
(3.330D)
Service
(3.330E)
Competition
(3.330F)
Other
(3.340)
Integrate customer dynamics with company economic factors
Determine mapping between important customer value segments and value elements with corresponding, crucial company economic factors
Store and make a list of such important relationships between customer value elements and company economic factors

Fig. 4

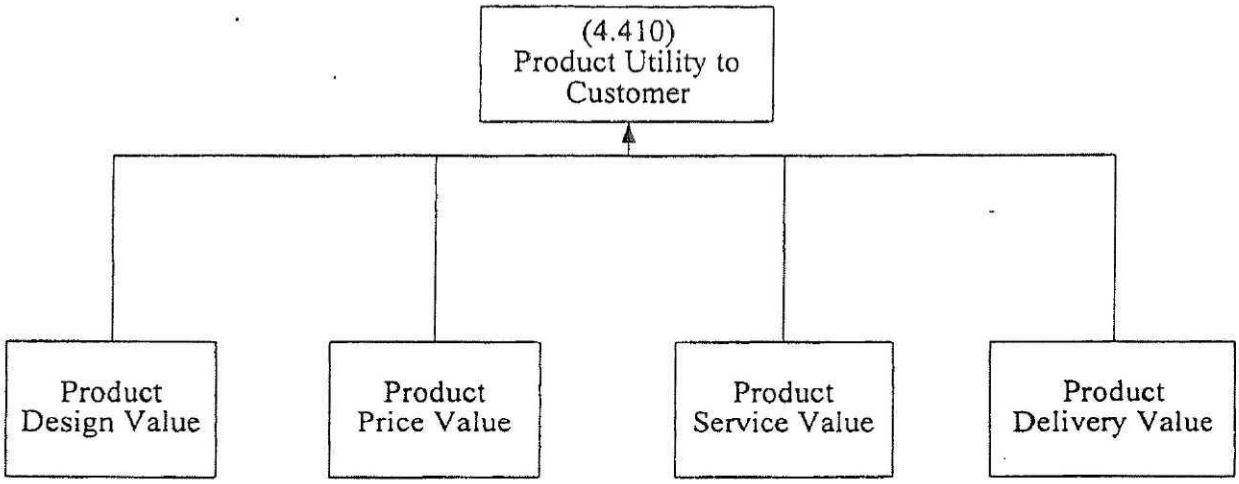

(4.410)
Product Utility to Customer
Product Design Value
Product Price Value
Product Service Value
Product Delivery Value

Fig. 5

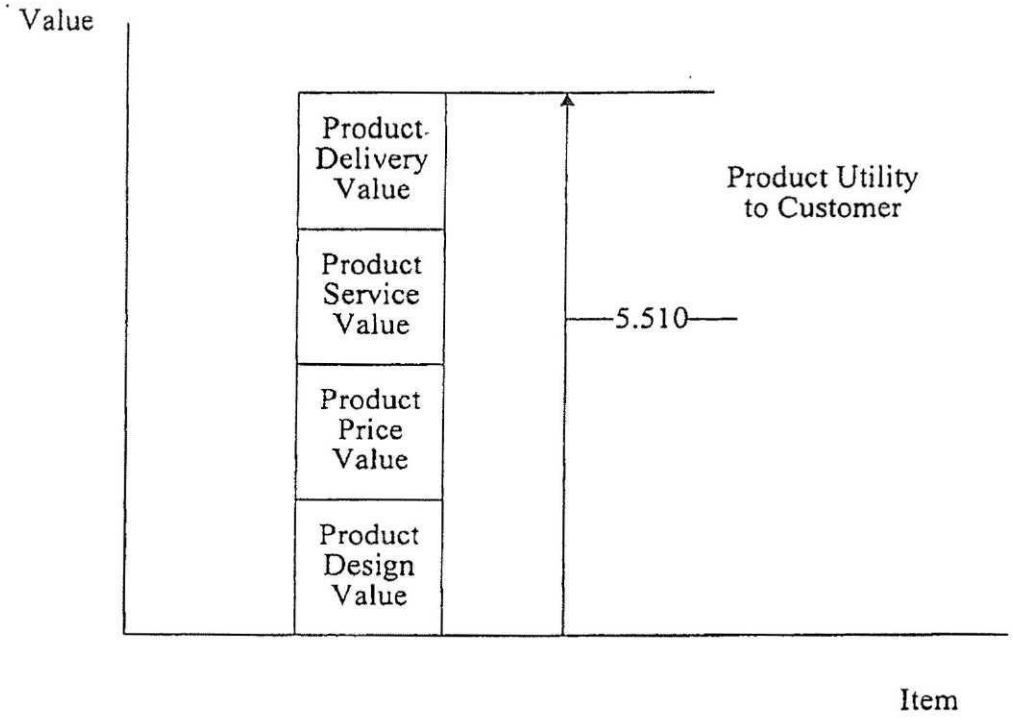

Value
Product Delivery Value
Product Service Value
Product Price Value
Product Design Value
Product Utility to Customer
5.510
Item

Fig. 6

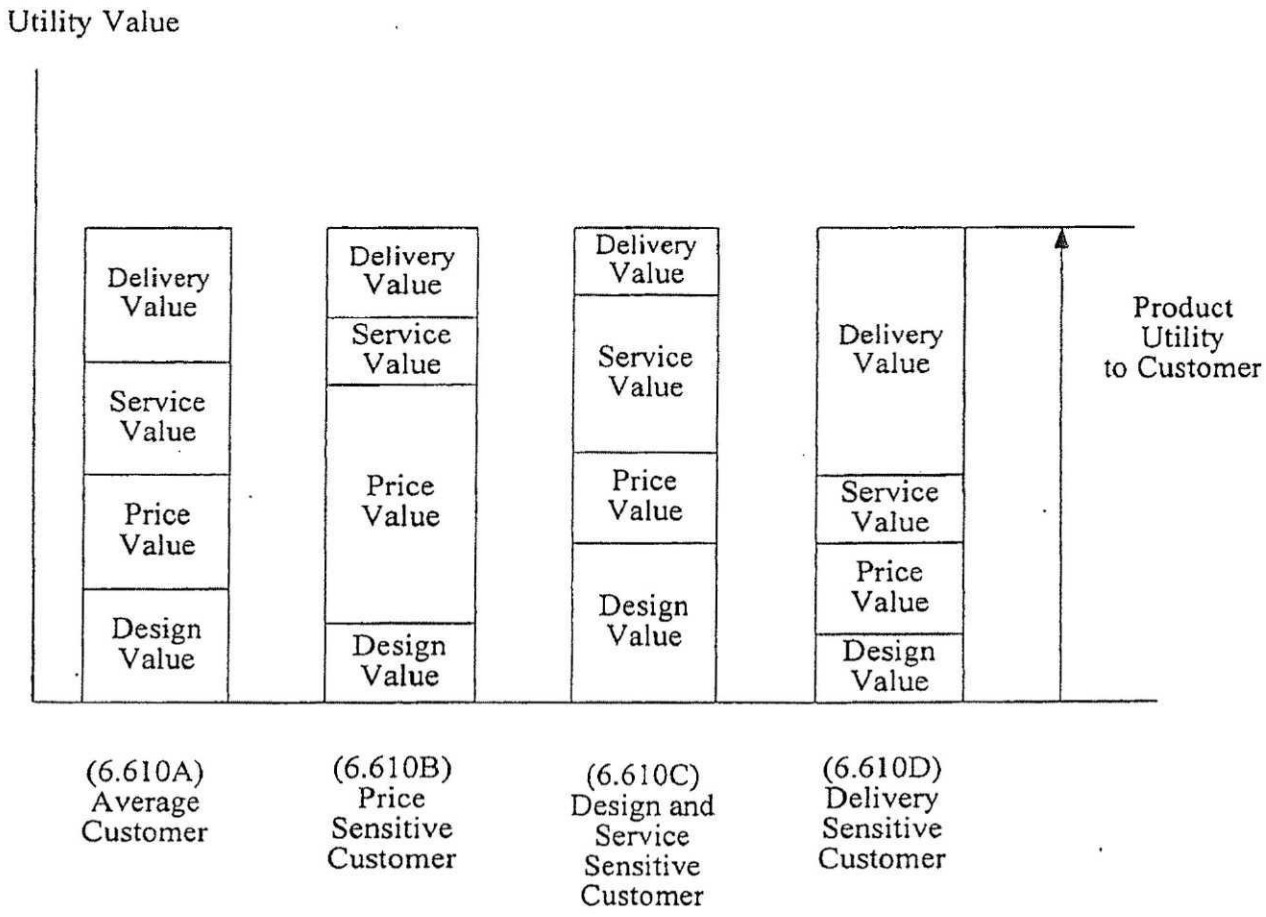

Utility Value
Delivery Value
Service Value
Price Value
Design Value
Delivery Value
Service Value
Price Value
Design Value
Delivery Value
Service Value
Price Value
Design Value
Delivery Value
Service Value
Price Value
Design Value
Product Utility to Customer
(6.610A) Average Customer
(6.610B) Price Sensitive Customer
(6.610C) Design and Service Sensitive Customer
(6.610D) Delivery Sensitive Customer

Fig. 7

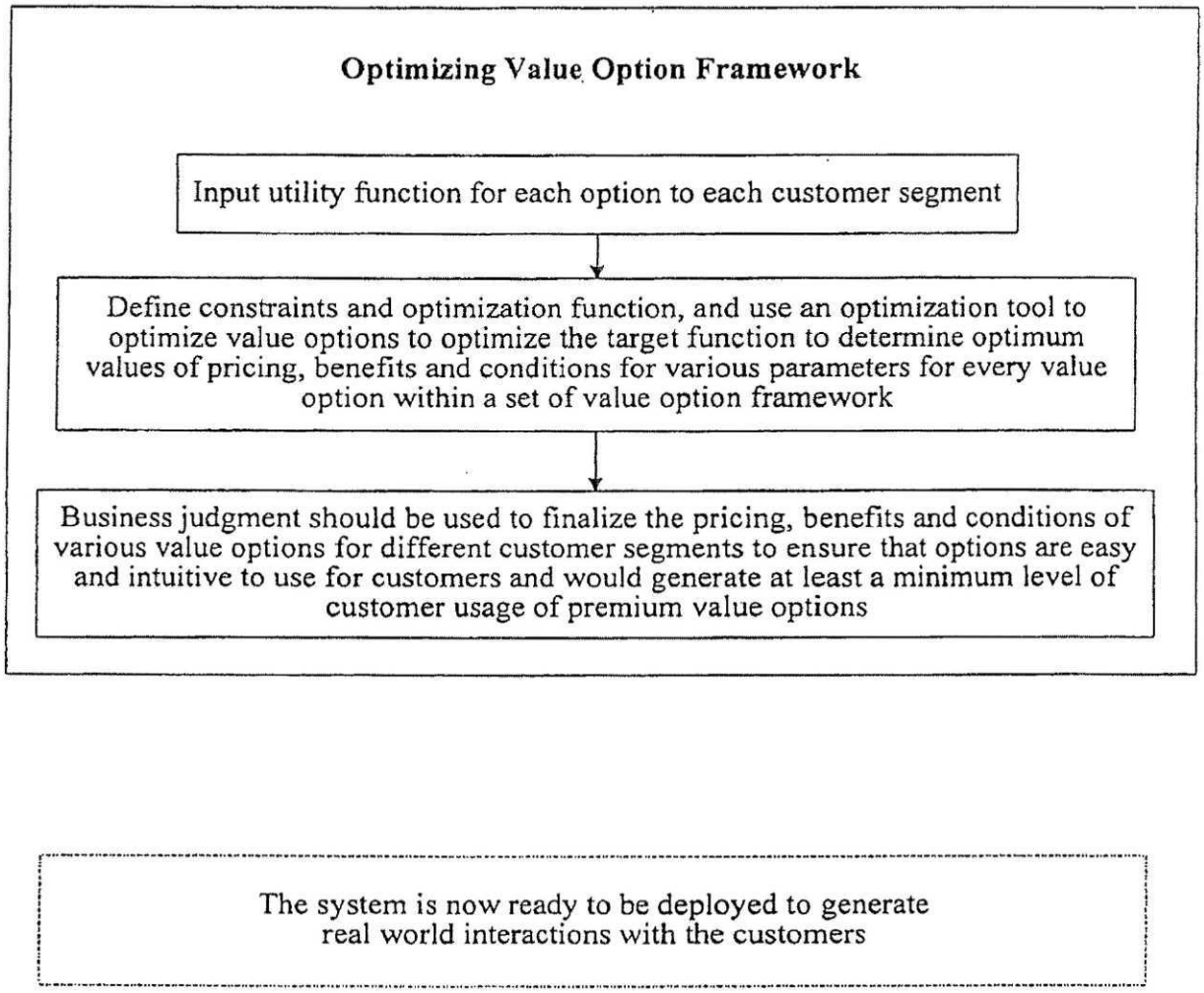

Optimizing Value Option Framework
Input utility function for each option to each customer segment
Define constraints and optimization function, and use an optimization tool to optimize value options to optimize the target function to determine optimum values of pricing, benefits and conditions for various parameters for every value option within a set of value option framework
Business judgment should be used to finalize the pricing, benefits and conditions of various value options for different customer segments to ensure that options are easy and intuitive to use for customers and would generate at least a minimum level of customer usage of premium value options
The system is now ready to be deployed to generate real world interactions with the customers

Fig. 8

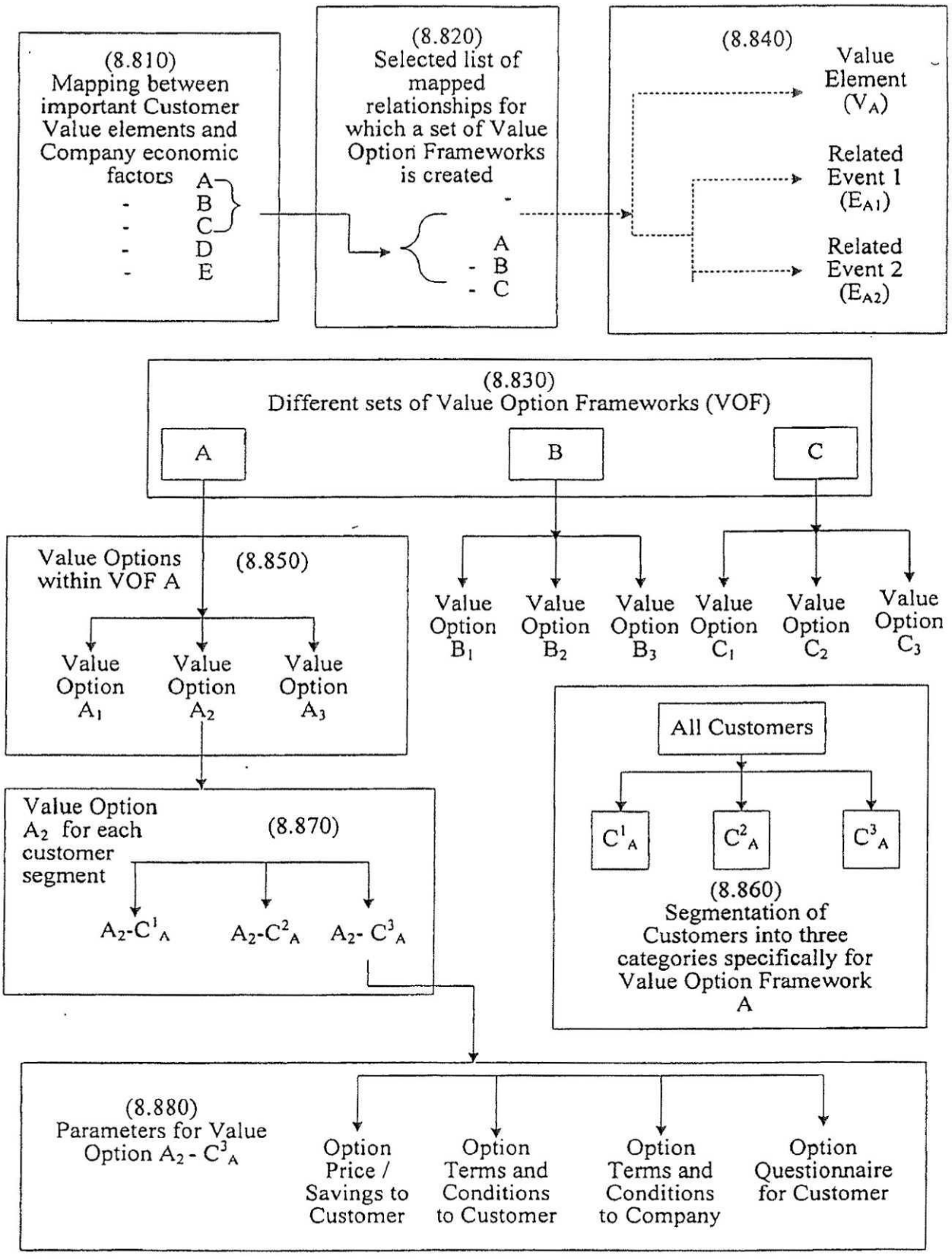

(8.810)
Mapping between important Customer Value elements and Company economic factors
A
B
C
D
E
(8.820)
Selected list of mapped relationships for which a set of Value Option Frameworks is created
A
B
C
(8.840)
Value Element (V_A)
Related Event 1 (E_{A1})
Related Event 2 (E_{A2})
(8.830)
Different sets of Value Option Frameworks (VOF)
A
B
C
Value Options within VOF A
(8.850)
Value Option A_1
Value Option A_2
Value Option A_3
Value Option B_1
Value Option B_2
Value Option B_3
Value Option C_1
Value Option C_2
Value Option C_3
All Customers
C^1_A
C^2_A
C^3_A
(8.860)
Segmentation of Customers into three categories specifically for Value Option Framework A
Value Option A_2 for each customer segment
(8.870)
A_2-C^1_A
A_2-C^2_A
A_2- C^3_A
(8.880)
Parameters for Value Option A_2 - C^3_A
Option Price / Savings to Customer
Option Terms and Conditions to Customer
Option Terms and Conditions to Company
Option Questionnaire for Customer

Fig. 9

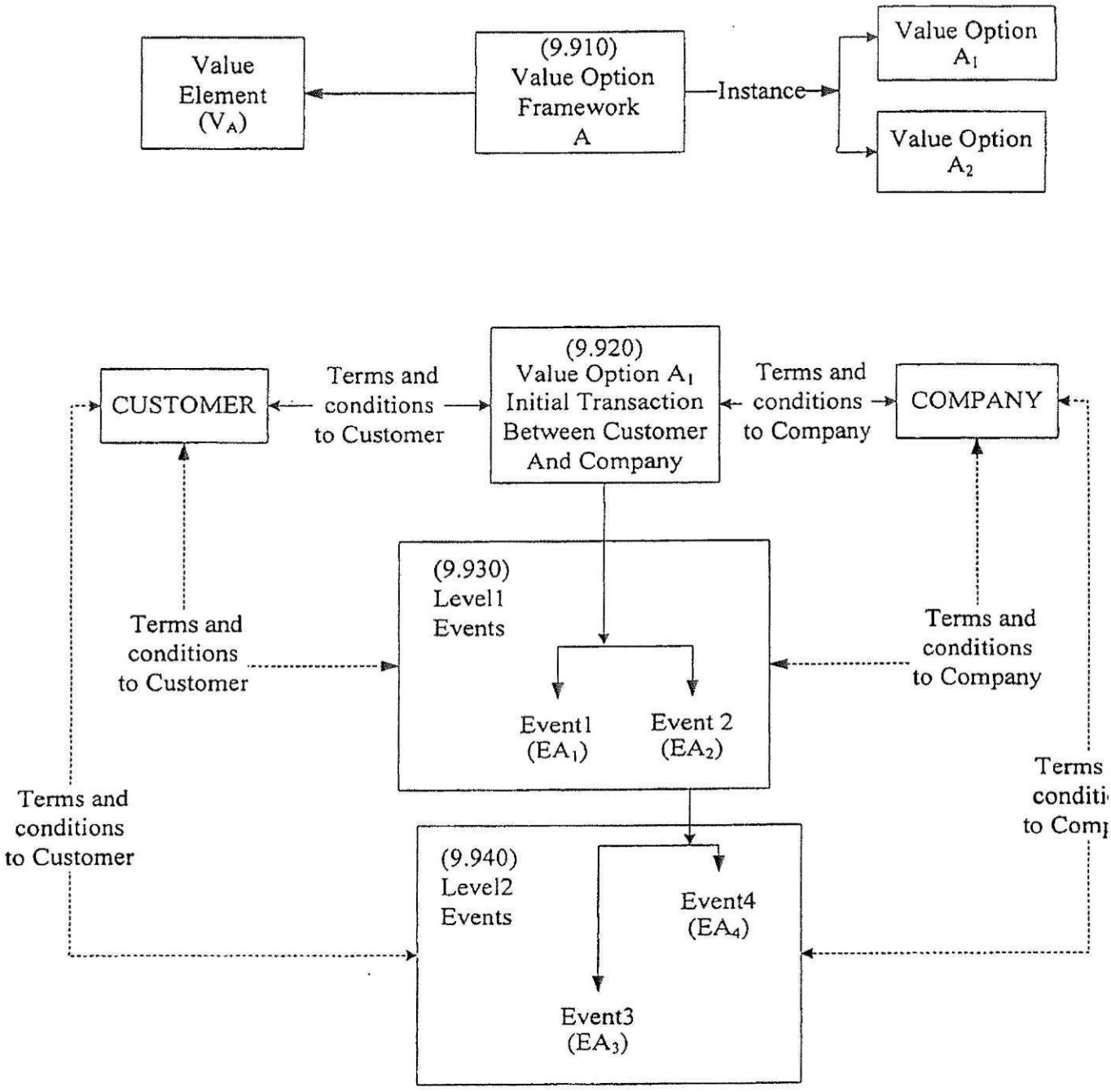

Value Element (V_A)
(9.910) Value Option Framework A
Instance
Value Option A_1
Value Option A_2
CUSTOMER
Terms and conditions to Customer
(9.920) Value Option A_1 Initial Transaction Between Customer And Company
Terms and conditions to Company
COMPANY
(9.930) Level1 Events
Event1 (EA_1)
Event 2 (EA_2)
Terms and conditions to Customer
Terms and conditions to Company
Terms and conditions to Customer
Terms conditi to Comp
(9.940) Level2 Events
Event4 (EA_4)
Event3 (EA_3)

Fig. 10

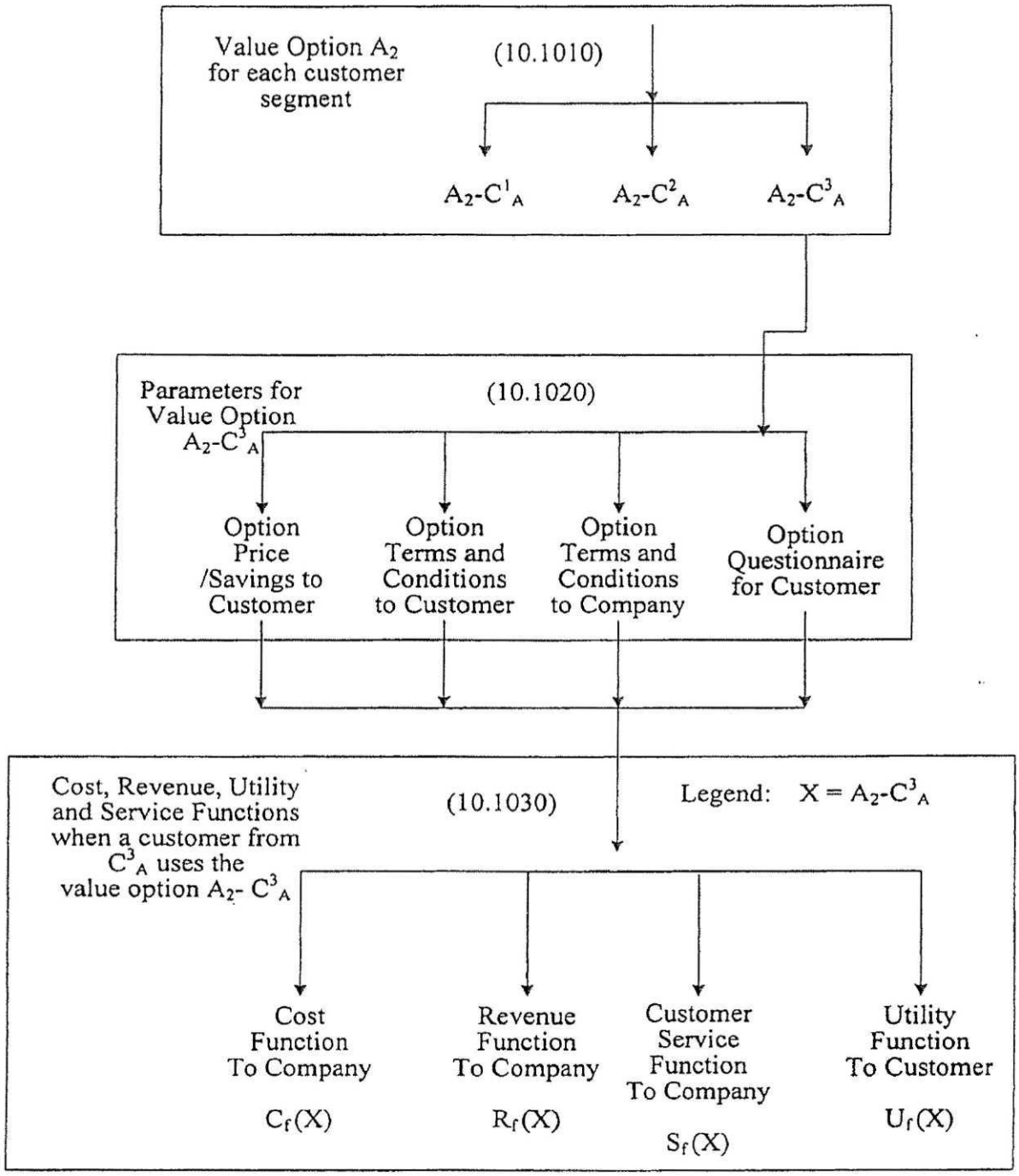

Value Option A_2 for each customer segment
(10.1010)
A_2-C^1_A
A_2-C^2_A
A_2-C^3_A
Parameters for Value Option A_2-C^3_A
(10.1020)
Option Price /Savings to Customer
Option Terms and Conditions to Customer
Option Terms and Conditions to Company
Option Questionnaire for Customer
Cost, Revenue, Utility and Service Functions when a customer from C^3_A uses the value option A_2- C^3_A
(10.1030)
Legend: $X = A_2$-C^3_A
Cost Function To Company
$C_f(X)$
Revenue Function To Company
$R_f(X)$
Customer Service Function To Company
$S_f(X)$
Utility Function To Customer
$U_f(X)$

Fig. 11

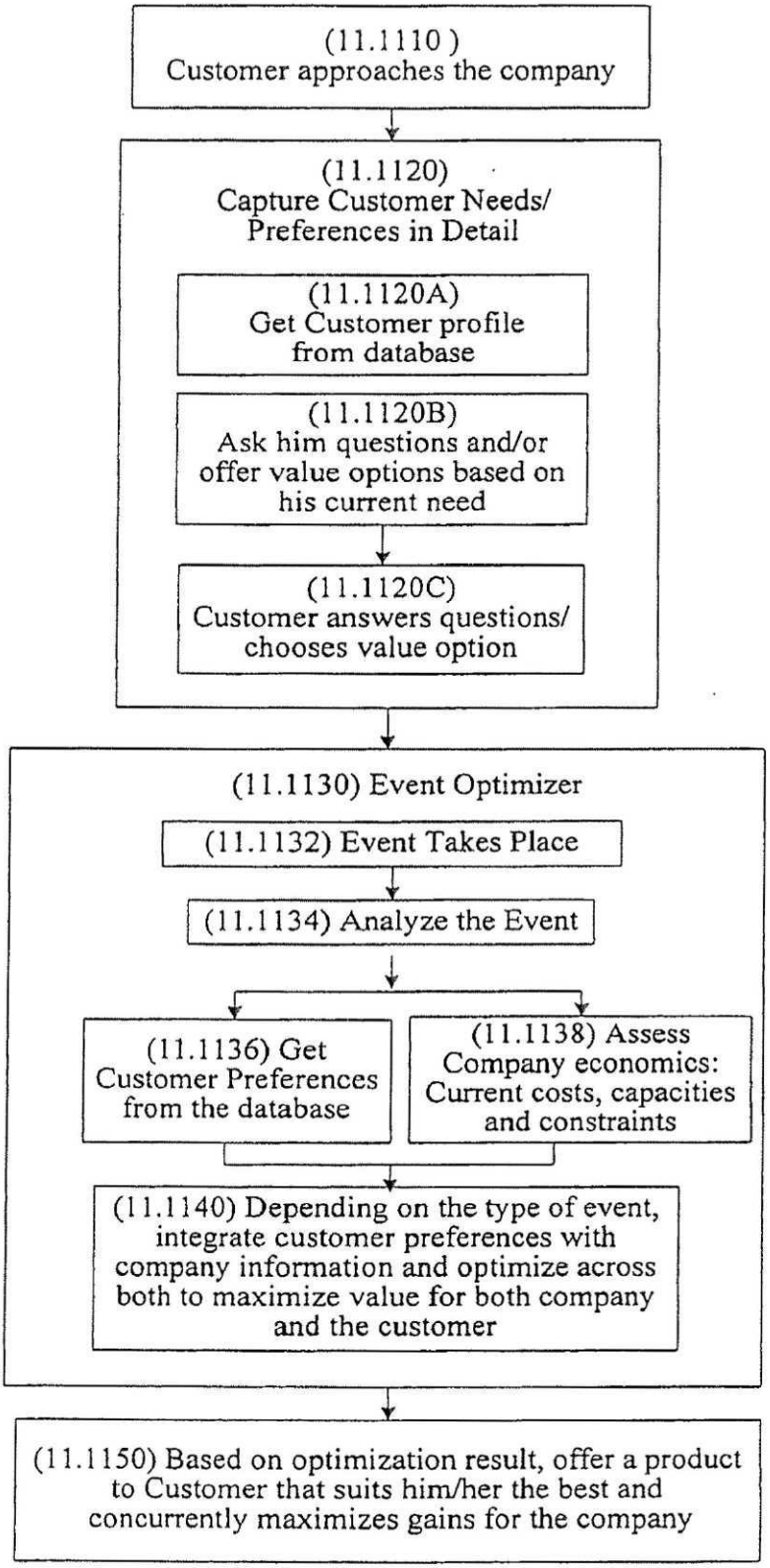

(11.1110)
Customer approaches the company
(11.1120)
Capture Customer Needs/
Preferences in Detail
(11.1120A)
Get Customer profile
from database
(11.1120B)
Ask him questions and/or
offer value options based on
his current need
(11.1120C)
Customer answers questions/
chooses value option
(11.1130) Event Optimizer
(11.1132) Event Takes Place
(11.1134) Analyze the Event
(11.1136) Get
Customer Preferences
from the database
(11.1138) Assess
Company economics:
Current costs, capacities
and constraints
(11.1140) Depending on the type of event,
integrate customer preferences with
company information and optimize across
both to maximize value for both company
and the customer
(11.1150) Based on optimization result, offer a product
to Customer that suits him/her the best and
concurrently maximizes gains for the company

Fig. 12

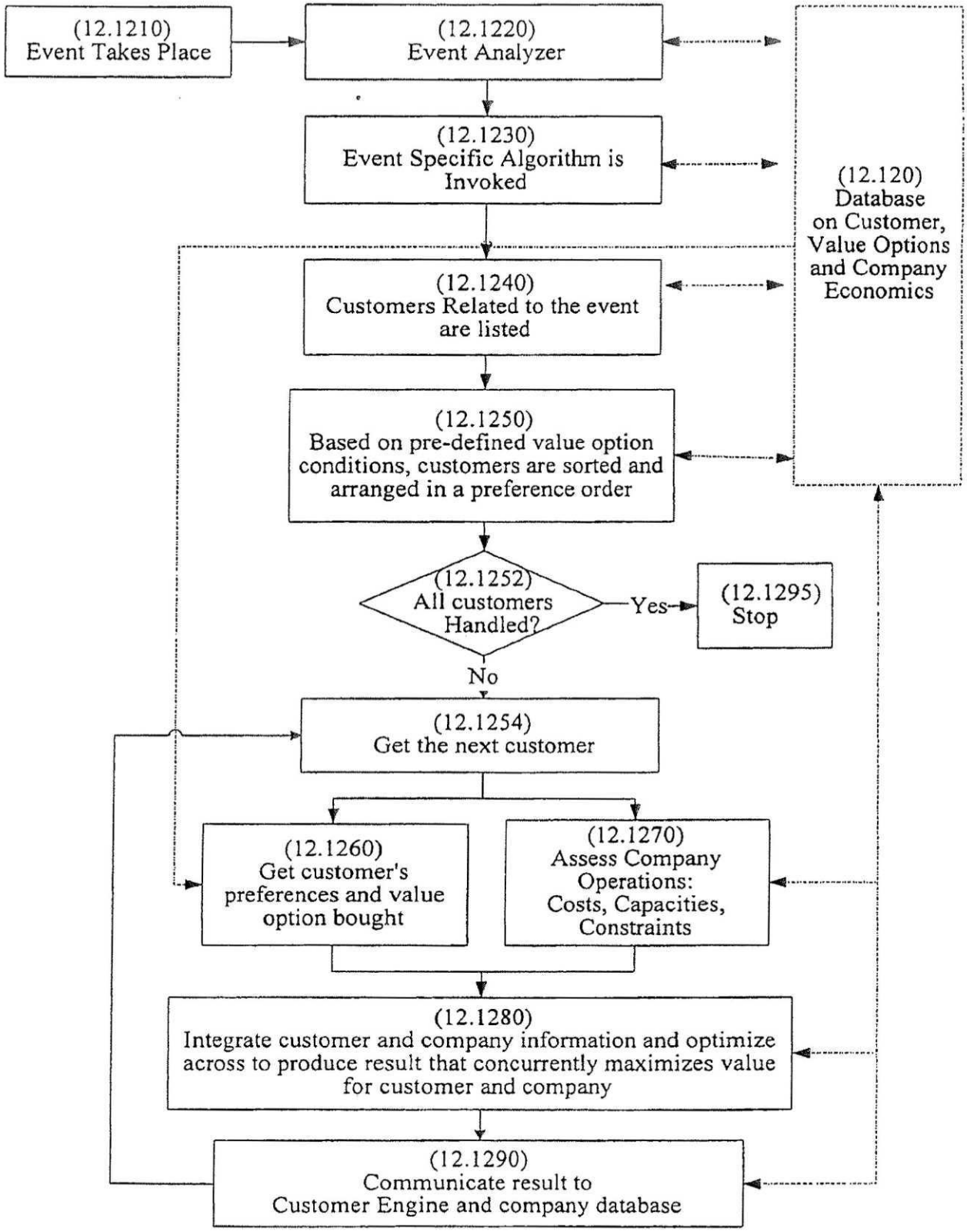

(12.1210)
Event Takes Place
(12.1220)
Event Analyzer
(12.1230)
Event Specific Algorithm is Invoked
(12.120)
Database on Customer, Value Options and Company Economics
(12.1240)
Customers Related to the event are listed
(12.1250)
Based on pre-defined value option conditions, customers are sorted and arranged in a preference order
(12.1252)
All customers Handled?
Yes
(12.1295)
Stop
No
(12.1254)
Get the next customer
(12.1260)
Get customer's preferences and value option bought
(12.1270)
Assess Company Operations: Costs, Capacities, Constraints
(12.1280)
Integrate customer and company information and optimize across to produce result that concurrently maximizes value for customer and company
(12.1290)
Communicate result to Customer Engine and company database

Fig. 13

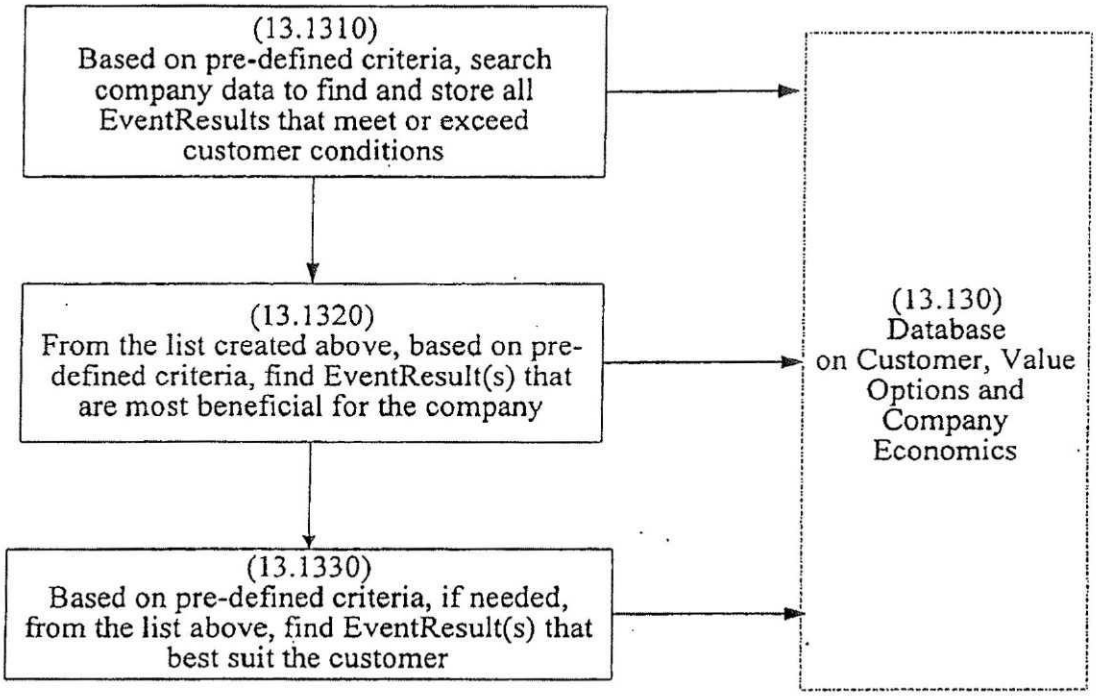

(13.1310)
Based on pre-defined criteria, search company data to find and store all EventResults that meet or exceed customer conditions
(13.1320)
From the list created above, based on pre-defined criteria, find EventResult(s) that are most beneficial for the company
(13.1330)
Based on pre-defined criteria, if needed, from the list above, find EventResult(s) that best suit the customer
(13.130)
Database
on Customer, Value Options and Company Economics

Fig. 13A

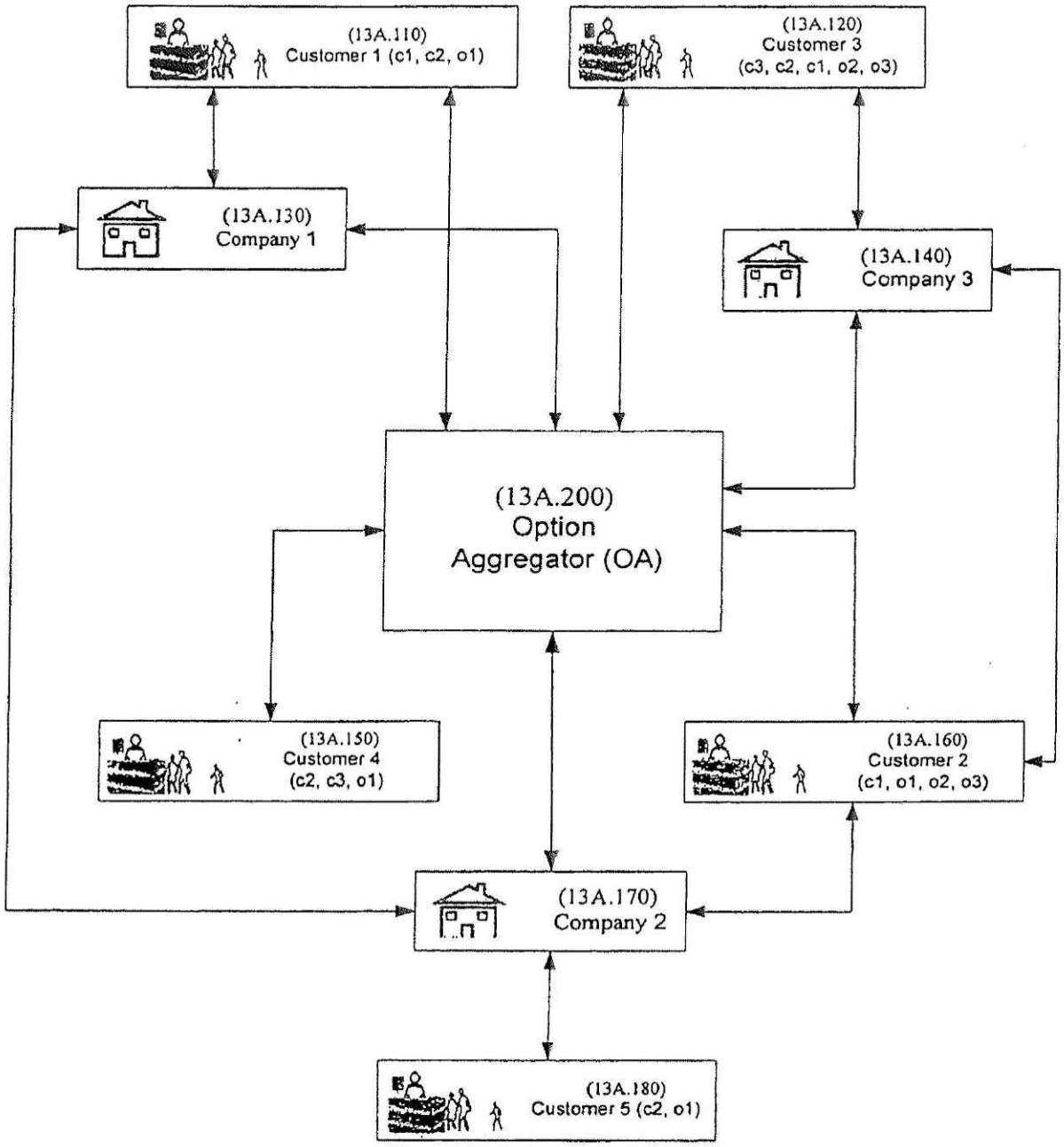

(13A.110)
Customer 1 (c1, c2, o1)
(13A.120)
Customer 3
(c3, c2, c1, o2, o3)
(13A.130)
Company 1
(13A.140)
Company 3
(13A.200)
Option
Aggregator (OA)
(13A.150)
Customer 4
(c2, c3, o1)
(13A.160)
Customer 2
(c1, o1, o2, o3)
(13A.170)
Company 2
(13A.180)
Customer 5 (c2, o1)

Fig. 13B

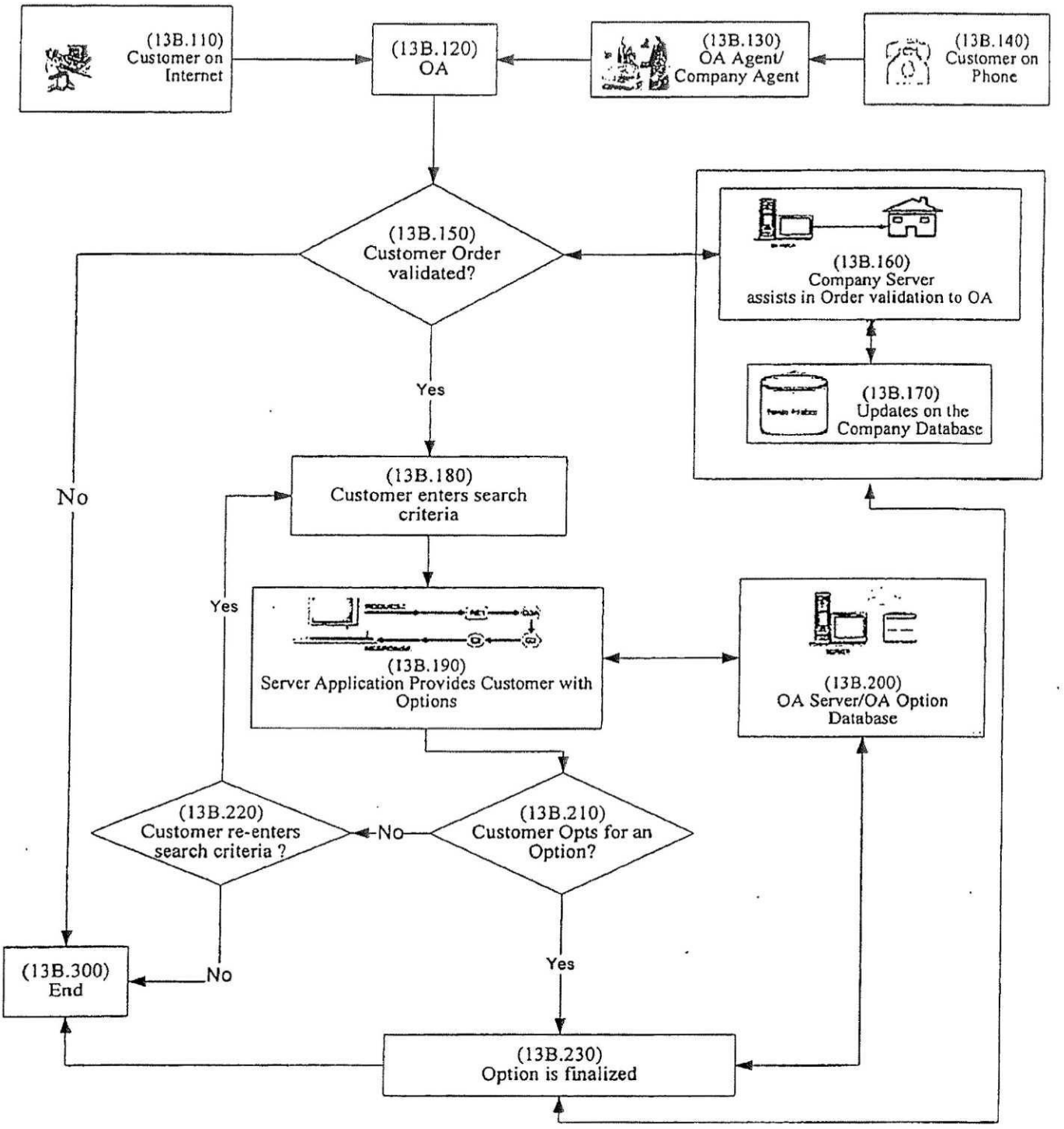

(13B.110) Customer on Internet
(13B.120) OA
(13B.130) OA Agent/ Company Agent
(13B.140) Customer on Phone
(13B.150) Customer Order validated?
(13B.160) Company Server assists in Order validation to OA
(13B.170) Updates on the Company Database
Yes
No
(13B.180) Customer enters search criteria
Yes
(13B.190) Server Application Provides Customer with Options
(13B.200) OA Server/OA Option Database
(13B.210) Customer Opts for an Option?
No
(13B.220) Customer re-enters search criteria ?
No
(13B.300) End
Yes
(13B.230) Option is finalized

Fig. 13C

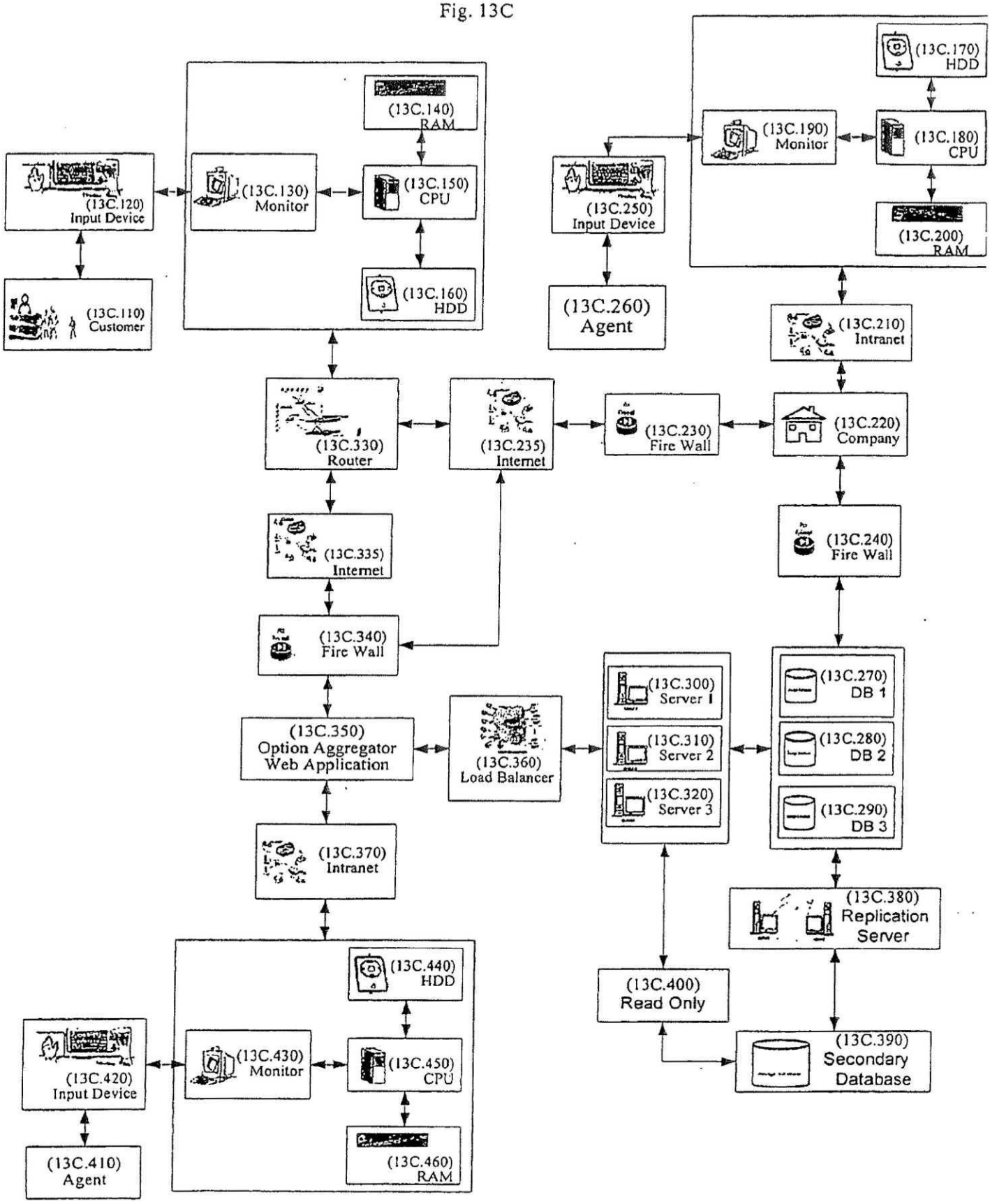

(13C.110) Customer
(13C.120) Input Device
(13C.130) Monitor
(13C.140) RAM
(13C.150) CPU
(13C.160) HDD
(13C.170) HDD
(13C.180) CPU
(13C.190) Monitor
(13C.200) RAM
(13C.210) Intranet
(13C.220) Company
(13C.230) Fire Wall
(13C.235) Internet
(13C.240) Fire Wall
(13C.250) Input Device
(13C.260) Agent
(13C.270) DB 1
(13C.280) DB 2
(13C.290) DB 3
(13C.300) Server 1
(13C.310) Server 2
(13C.320) Server 3
(13C.330) Router
(13C.335) Internet
(13C.340) Fire Wall
(13C.350) Option Aggregator Web Application
(13C.360) Load Balancer
(13C.370) Intranet
(13C.380) Replication Server
(13C.390) Secondary Database
(13C.400) Read Only
(13C.410) Agent
(13C.420) Input Device
(13C.430) Monitor
(13C.440) HDD
(13C.450) CPU
(13C.460) RAM

Fig. 13D

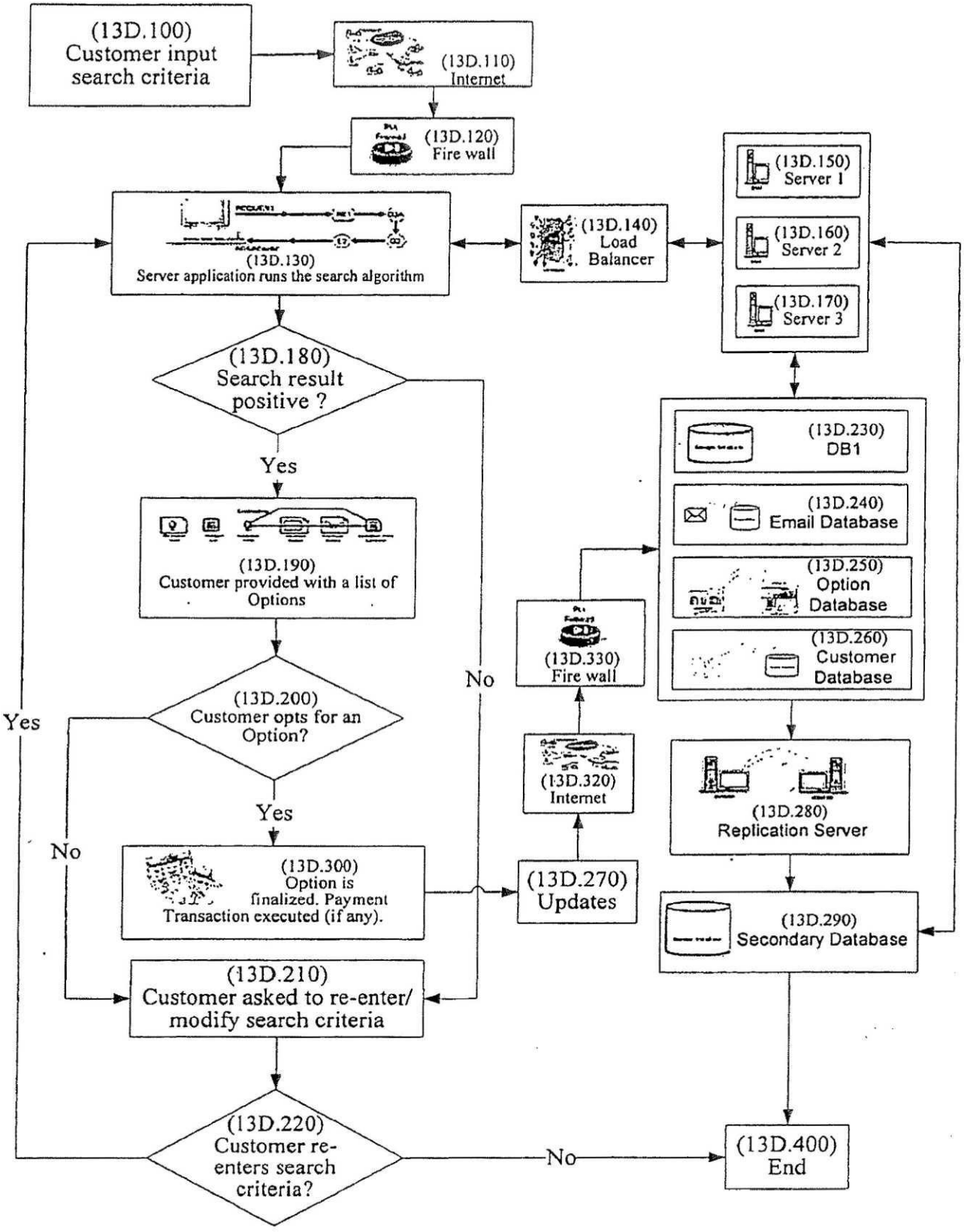

(13D.100)
Customer input search criteria
(13D.110)
Internet
(13D.120)
Fire wall
(13D.130)
Server application runs the search algorithm
(13D.140)
Load Balancer
(13D.150)
Server 1
(13D.160)
Server 2
(13D.170)
Server 3
(13D.180)
Search result positive ?
Yes
No
(13D.190)
Customer provided with a list of Options
(13D.200)
Customer opts for an Option?
Yes
No
(13D.300)
Option is finalized. Payment Transaction executed (if any).
(13D.210)
Customer asked to re-enter/ modify search criteria
(13D.220)
Customer re-enters search criteria?
Yes
No
(13D.230)
DB1
(13D.240)
Email Database
(13D.250)
Option Database
(13D.260)
Customer Database
(13D.330)
Fire wall
(13D.320)
Internet
(13D.270)
Updates
(13D.280)
Replication Server
(13D.290)
Secondary Database
(13D.400)
End

Fig. 13E

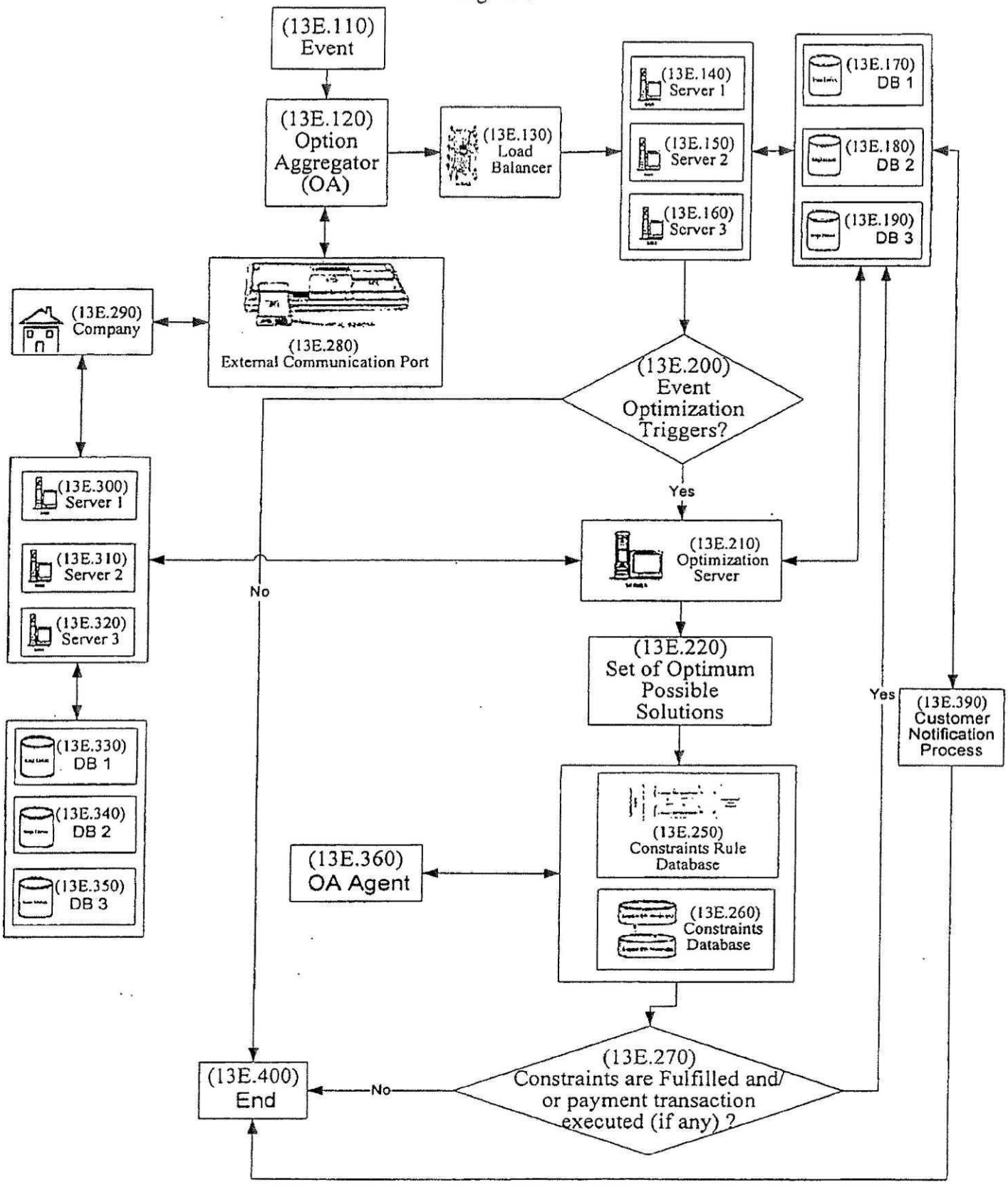

(13E.110) Event
(13E.120) Option Aggregator (OA)
(13E.130) Load Balancer
(13E.140) Server 1
(13E.150) Server 2
(13E.160) Server 3
(13E.170) DB 1
(13E.180) DB 2
(13E.190) DB 3
(13E.290) Company
(13E.280) External Communication Port
(13E.200) Event Optimization Triggers?
Yes
No
(13E.300) Server 1
(13E.310) Server 2
(13E.320) Server 3
(13E.210) Optimization Server
(13E.220) Set of Optimum Possible Solutions
(13E.330) DB 1
(13E.340) DB 2
(13E.350) DB 3
(13E.360) OA Agent
(13E.250) Constraints Rule Database
(13E.260) Constraints Database
(13E.270) Constraints are Fulfilled and/ or payment transaction executed (if any) ?
(13E.390) Customer Notification Process
(13E.400) End

Fig. 14

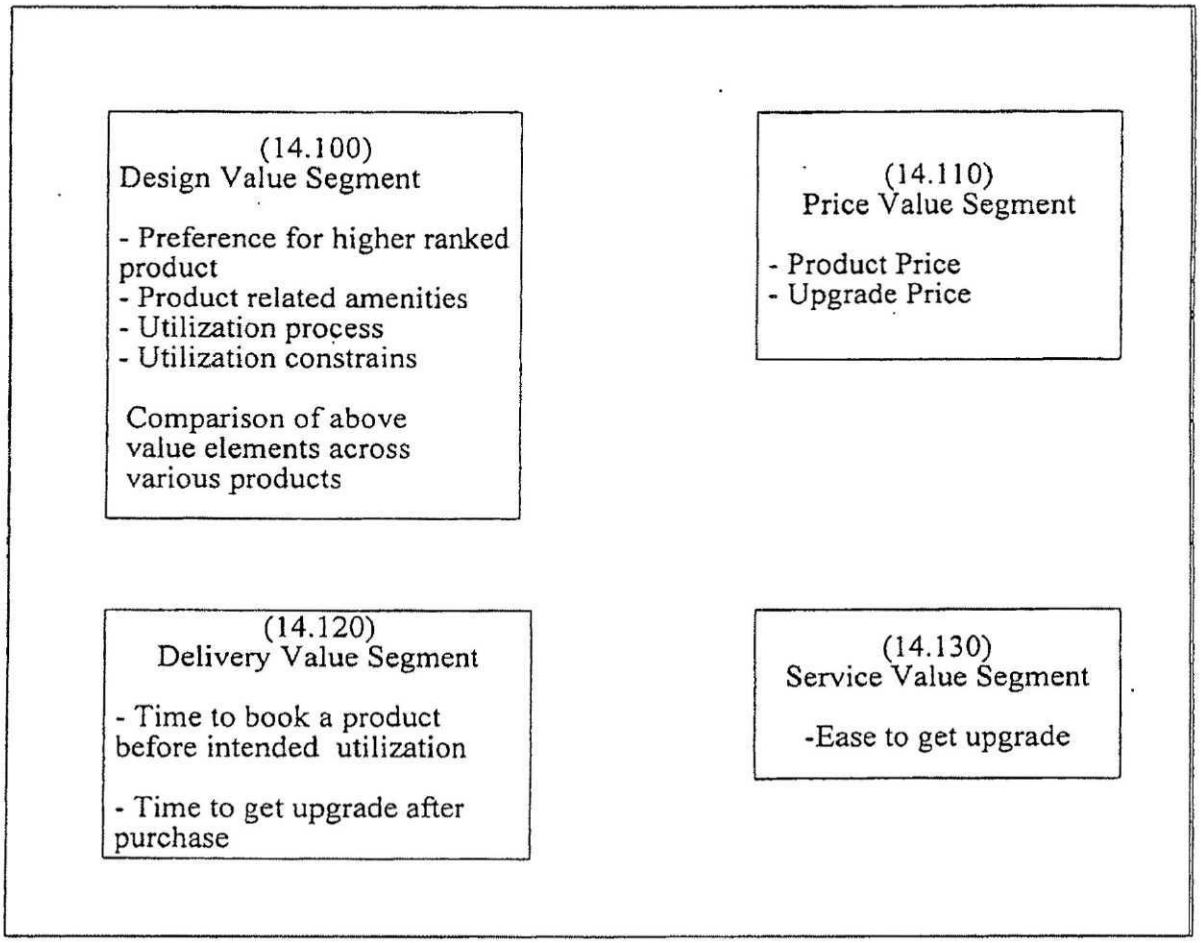

(14.100)
Design Value Segment
- Preference for higher ranked product
- Product related amenities
- Utilization process
- Utilization constrains
Comparison of above value elements across various products
(14.110)
Price Value Segment
- Product Price
- Upgrade Price
(14.120)
Delivery Value Segment
- Time to book a product before intended utilization
- Time to get upgrade after purchase
(14.130)
Service Value Segment
-Ease to get upgrade

Fig. 15

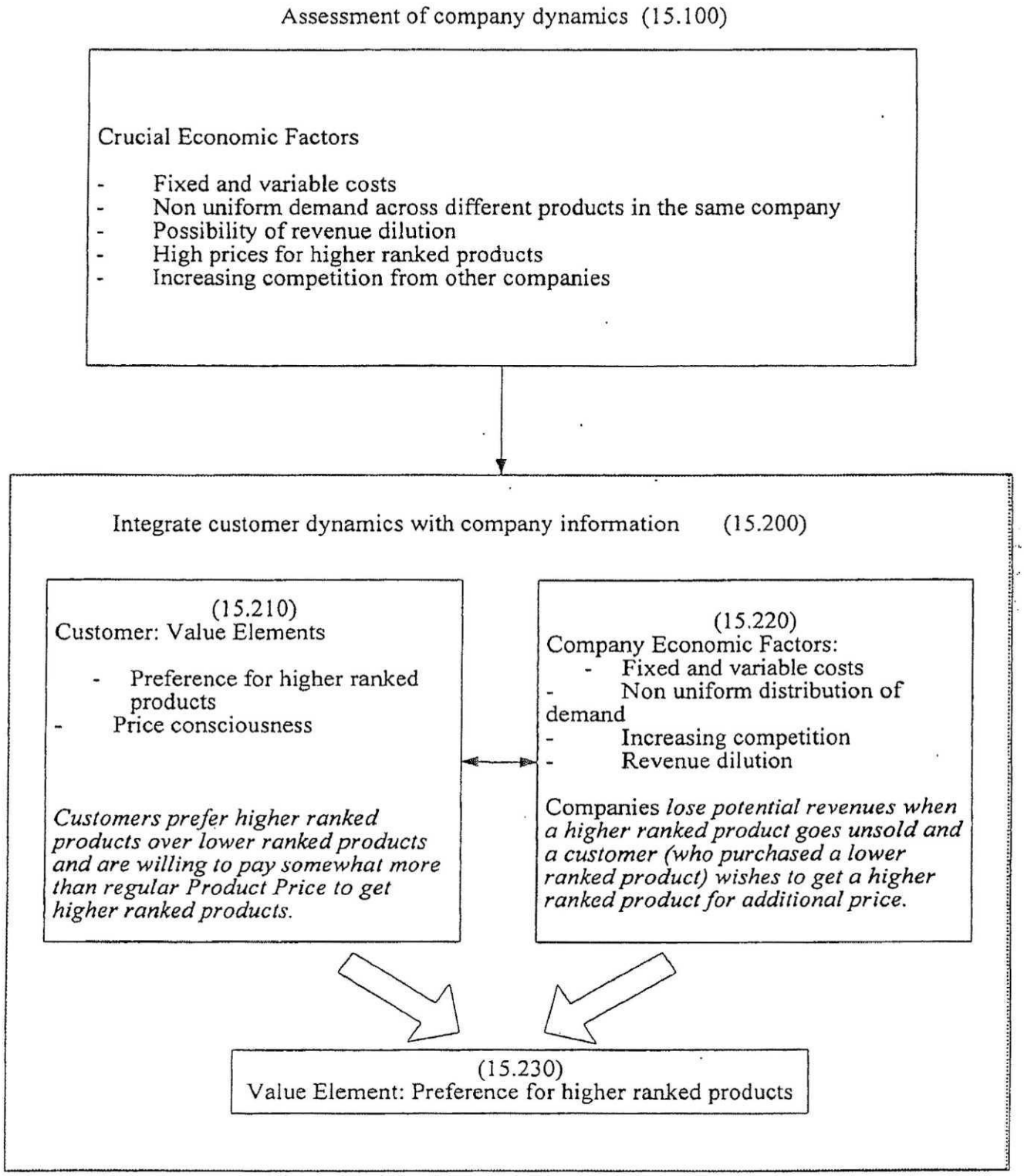

Assessment of company dynamics (15.100)
Crucial Economic Factors
- Fixed and variable costs
- Non uniform demand across different products in the same company
- Possibility of revenue dilution
- High prices for higher ranked products
- Increasing competition from other companies
Integrate customer dynamics with company information (15.200)
(15.210)
Customer: Value Elements
- Preference for higher ranked products
- Price consciousness
Customers prefer higher ranked products over lower ranked products and are willing to pay somewhat more than regular Product Price to get higher ranked products.
(15.220)
Company Economic Factors:
- Fixed and variable costs
- Non uniform distribution of demand
- Increasing competition
- Revenue dilution
Companies lose potential revenues when a higher ranked product goes unsold and a customer (who purchased a lower ranked product) wishes to get a higher ranked product for additional price.
(15.230)
Value Element: Preference for higher ranked products

Fig. 16

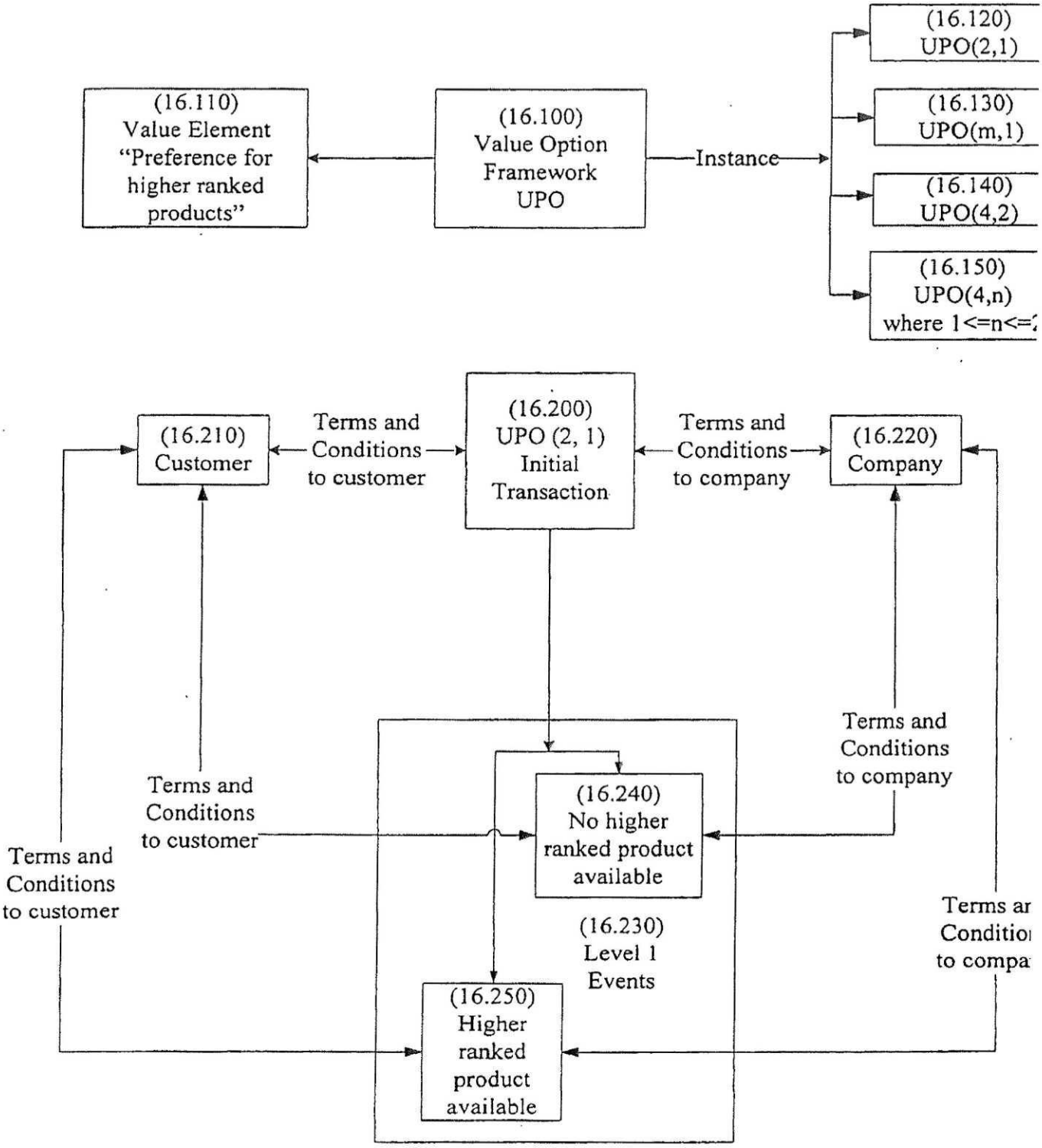

(16.110)
Value Element
"Preference for higher ranked products"
(16.100)
Value Option Framework UPO
Instance
(16.120)
UPO(2,1)
(16.130)
UPO(m,1)
(16.140)
UPO(4,2)
(16.150)
UPO(4,n)
where 1<=n<=
(16.210)
Customer
Terms and Conditions to customer
(16.200)
UPO (2, 1)
Initial Transaction
Terms and Conditions to company
(16.220)
Company
Terms and Conditions to customer
Terms and Conditions to customer
(16.240)
No higher ranked product available
Terms and Conditions to company
(16.230)
Level 1 Events
(16.250)
Higher ranked product available
Terms ar
Conditio
to compa

Fig. 17

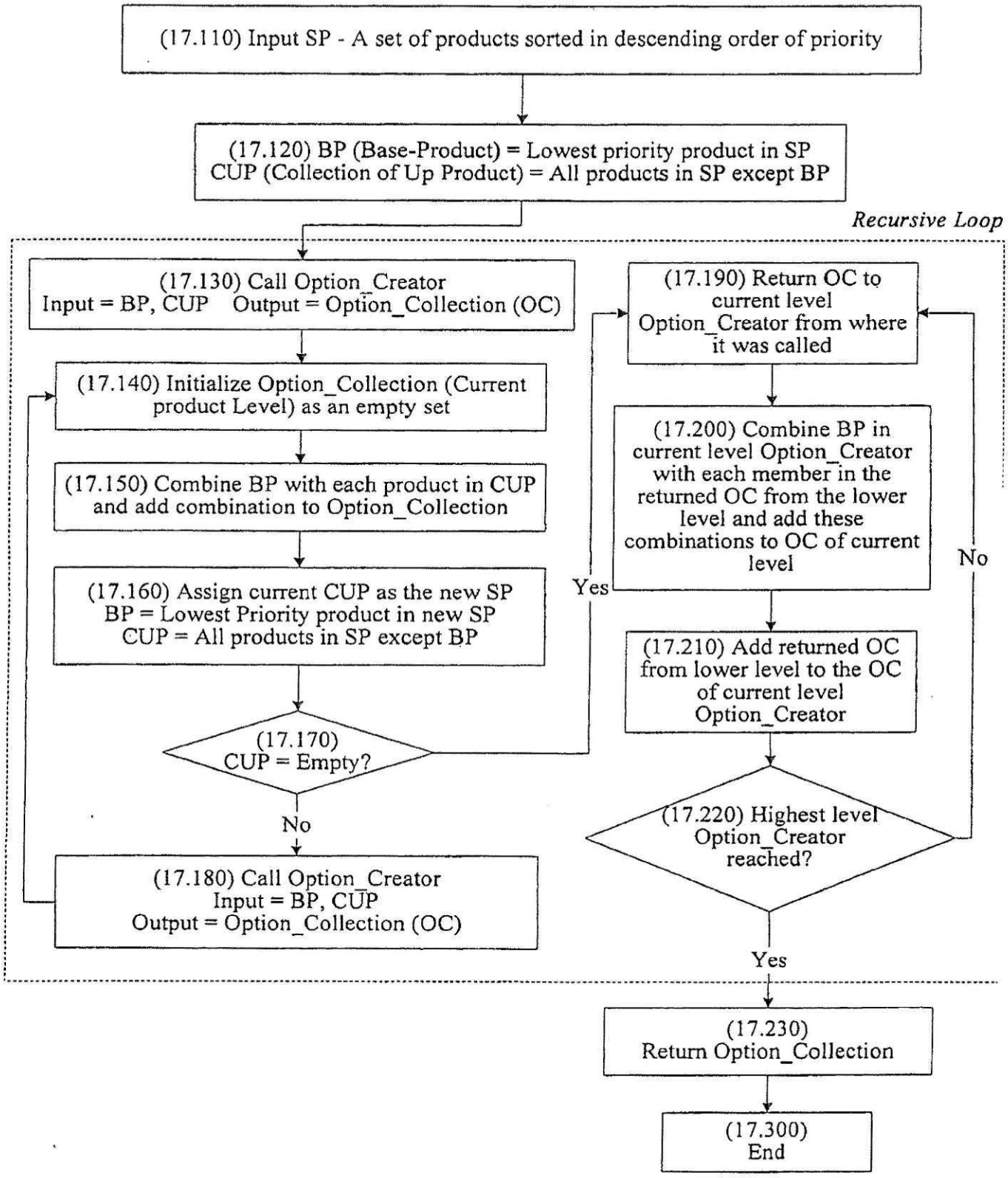

(17.110) Input SP - A set of products sorted in descending order of priority
(17.120) BP (Base-Product) = Lowest priority product in SP
CUP (Collection of Up Product) = All products in SP except BP
Recursive Loop
(17.130) Call Option_Creator
Input = BP, CUP Output = Option_Collection (OC)
(17.140) Initialize Option_Collection (Current product Level) as an empty set
(17.150) Combine BP with each product in CUP and add combination to Option_Collection
(17.160) Assign current CUP as the new SP
BP = Lowest Priority product in new SP
CUP = All products in SP except BP
(17.170)
CUP = Empty?
Yes
No
(17.180) Call Option_Creator
Input = BP, CUP
Output = Option_Collection (OC)
(17.190) Return OC to current level Option_Creator from where it was called
(17.200) Combine BP in current level Option_Creator with each member in the returned OC from the lower level and add these combinations to OC of current level
(17.210) Add returned OC from lower level to the OC of current level Option_Creator
(17.220) Highest level Option_Creator reached?
No
Yes
(17.230)
Return Option_Collection
(17.300)
End

Fig. 18

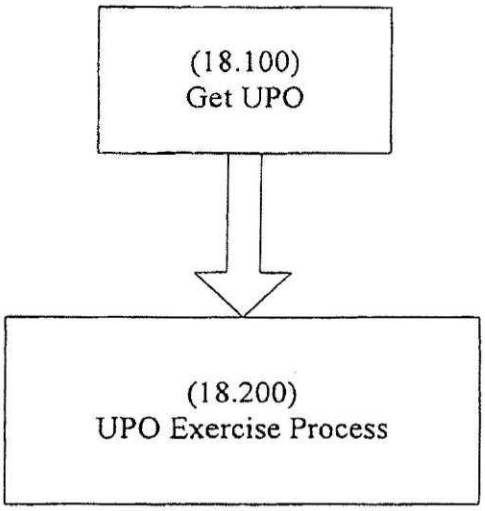


Fig. 19

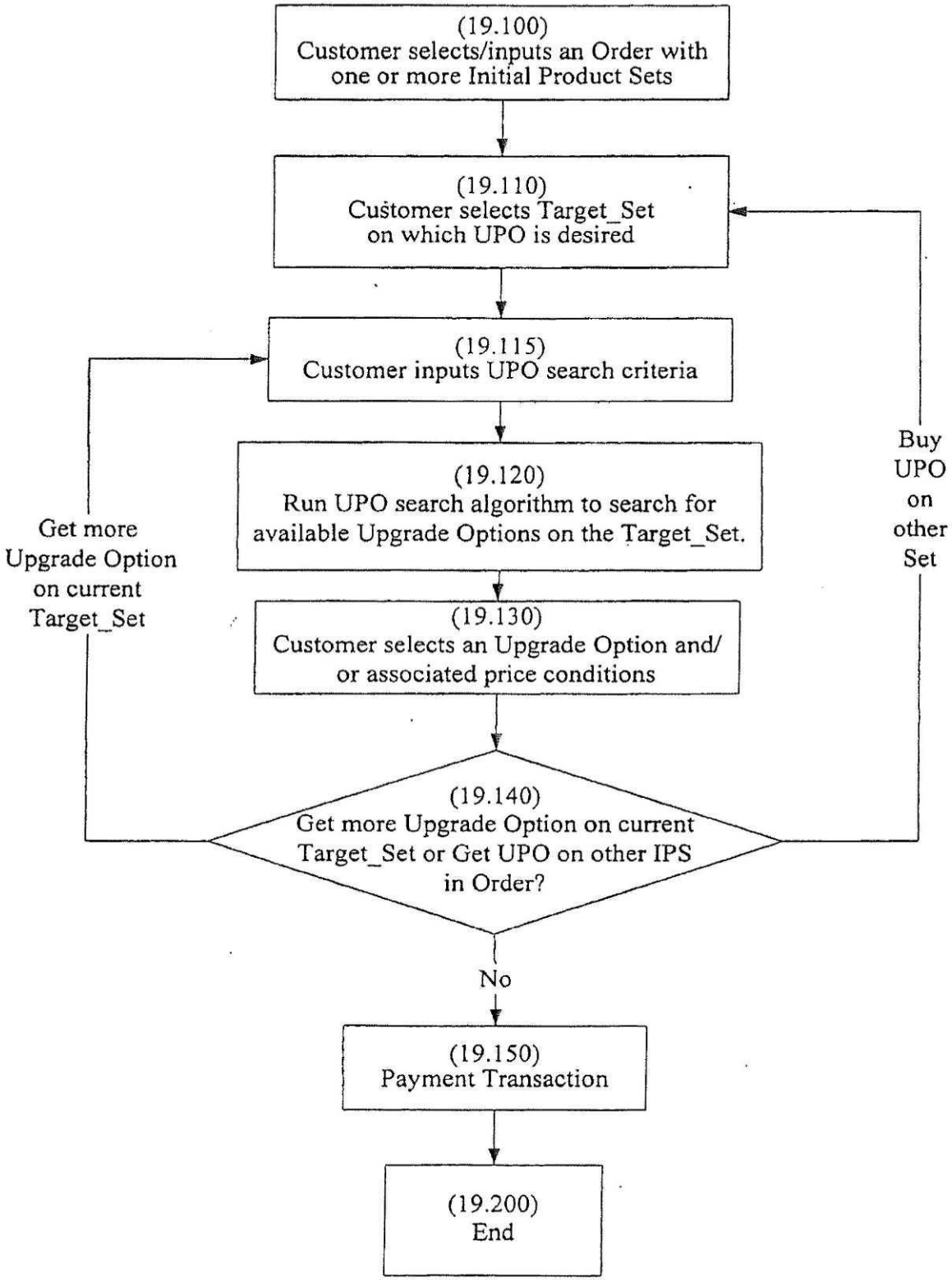

(19.100)
Customer selects/inputs an Order with one or more Initial Product Sets
(19.110)
Customer selects Target_Set on which UPO is desired
(19.115)
Customer inputs UPO search criteria
(19.120)
Run UPO search algorithm to search for available Upgrade Options on the Target_Set.
(19.130)
Customer selects an Upgrade Option and/ or associated price conditions
(19.140)
Get more Upgrade Option on current Target_Set or Get UPO on other IPS in Order?
Get more Upgrade Option on current Target_Set
Buy UPO on other Set
No
(19.150)
Payment Transaction
(19.200)
End

Fig. 20

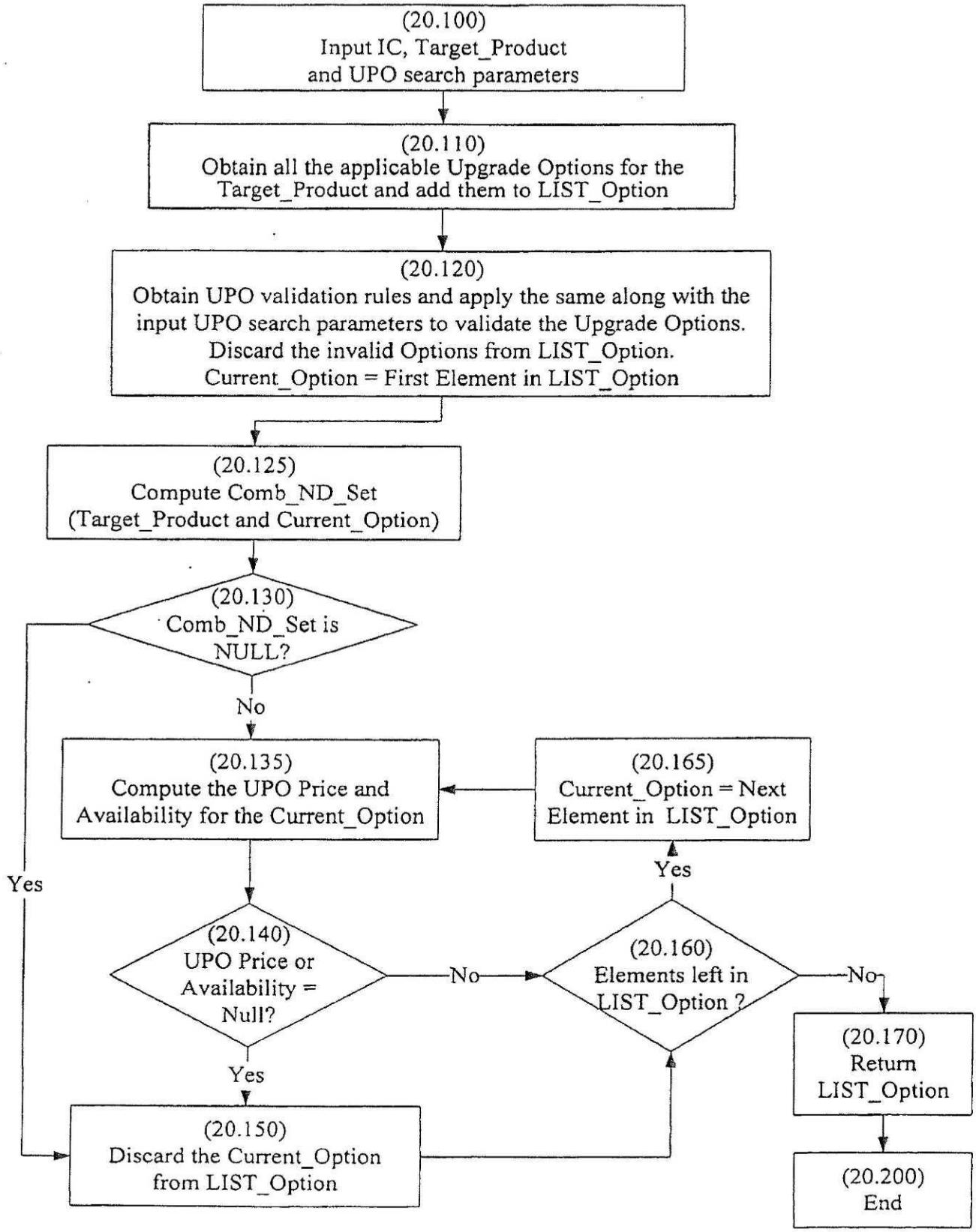

(20.100)
Input IC, Target_Product
and UPO search parameters
(20.110)
Obtain all the applicable Upgrade Options for the
Target_Product and add them to LIST_Option
(20.120)
Obtain UPO validation rules and apply the same along with the
input UPO search parameters to validate the Upgrade Options.
Discard the invalid Options from LIST_Option.
Current_Option = First Element in LIST_Option
(20.125)
Compute Comb_ND_Set
(Target_Product and Current_Option)
(20.130)
Comb_ND_Set is
NULL?
No
Yes
(20.135)
Compute the UPO Price and
Availability for the Current_Option
(20.165)
Current_Option = Next
Element in LIST_Option
Yes
(20.140)
UPO Price or
Availability =
Null?
No
(20.160)
Elements left in
LIST_Option ?
No
Yes
(20.150)
Discard the Current_Option
from LIST_Option
(20.170)
Return
LIST_Option
(20.200)
End

Fig. 21

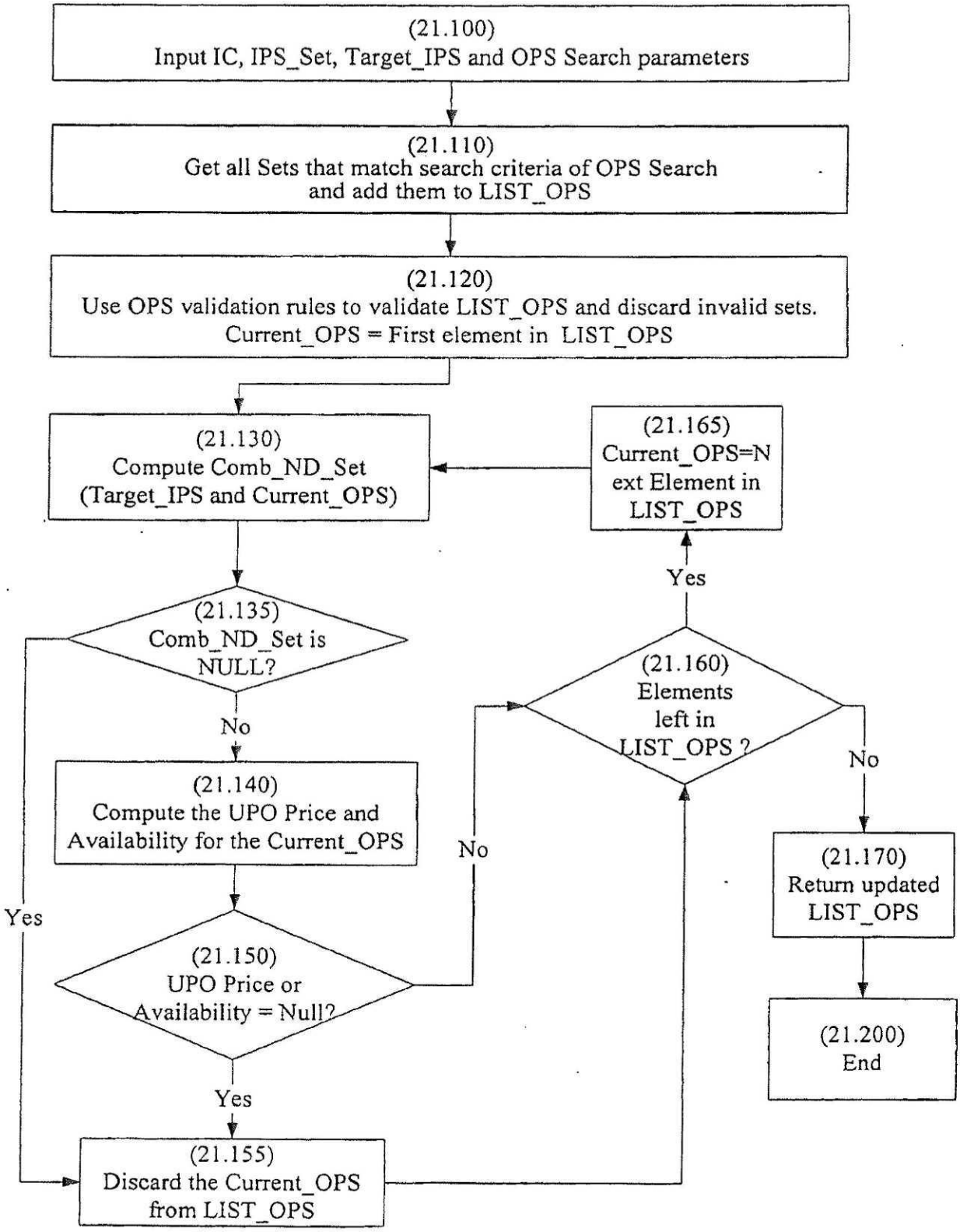

(21.100)
Input IC, IPS_Set, Target_IPS and OPS Search parameters
(21.110)
Get all Sets that match search criteria of OPS Search and add them to LIST_OPS
(21.120)
Use OPS validation rules to validate LIST_OPS and discard invalid sets.
Current_OPS = First element in LIST_OPS
(21.130)
Compute Comb_ND_Set (Target_IPS and Current_OPS)
(21.165)
Current_OPS=Next Element in LIST_OPS
(21.135)
Comb_ND_Set is NULL?
Yes
No
(21.140)
Compute the UPO Price and Availability for the Current_OPS
(21.160)
Elements left in LIST_OPS ?
No
Yes
(21.150)
UPO Price or Availability = Null?
No
Yes
(21.155)
Discard the Current_OPS from LIST_OPS
(21.170)
Return updated LIST_OPS
(21.200)
End

Fig. 22

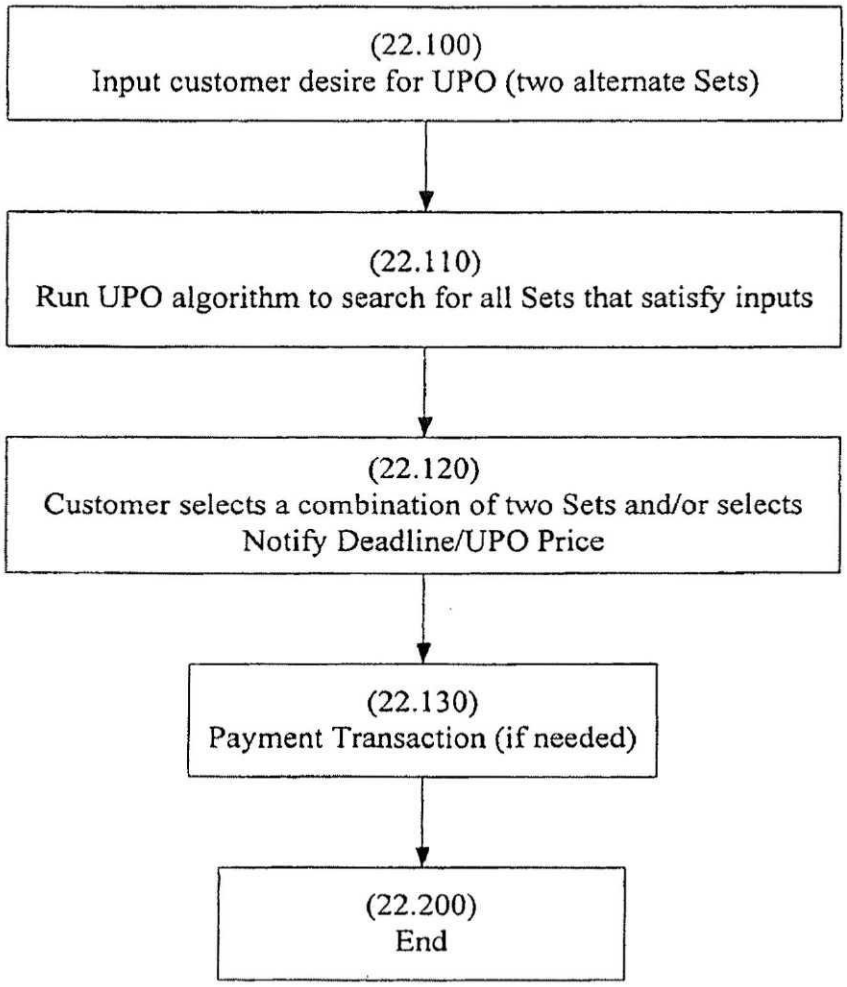

(22.100)
Input customer desire for UPO (two alternate Sets)
(22.110)
Run UPO algorithm to search for all Sets that satisfy inputs
(22.120)
Customer selects a combination of two Sets and/or selects
Notify Deadline/UPO Price
(22.130)
Payment Transaction (if needed)
(22.200)
End

Fig. 23

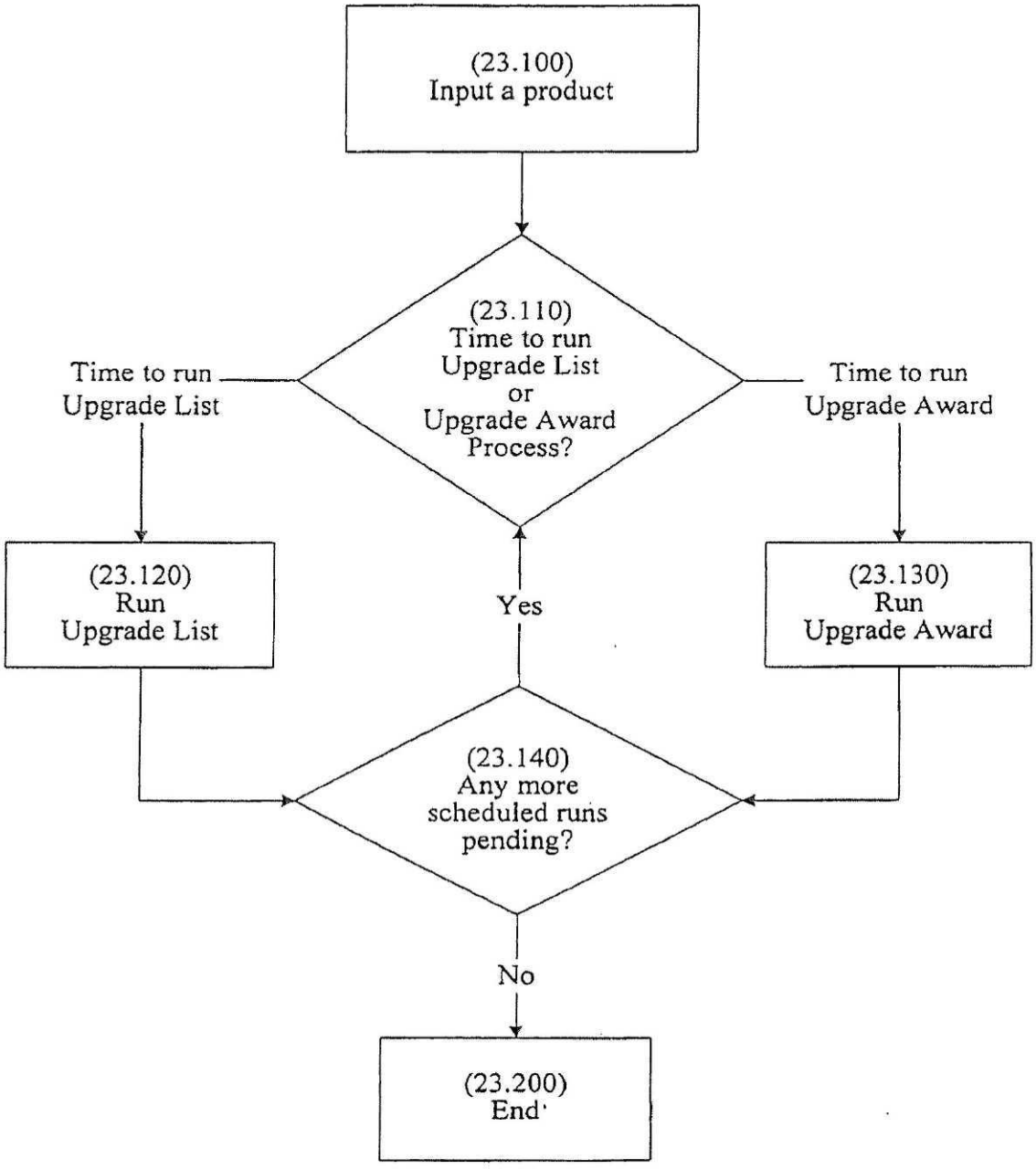

(23.100)
Input a product
(23.110)
Time to run
Upgrade List
or
Upgrade Award
Process?
Time to run
Upgrade List
Time to run
Upgrade Award
(23.120)
Run
Upgrade List
Yes
(23.130)
Run
Upgrade Award
(23.140)
Any more
scheduled runs
pending?
No
(23.200)
End

Fig. 24

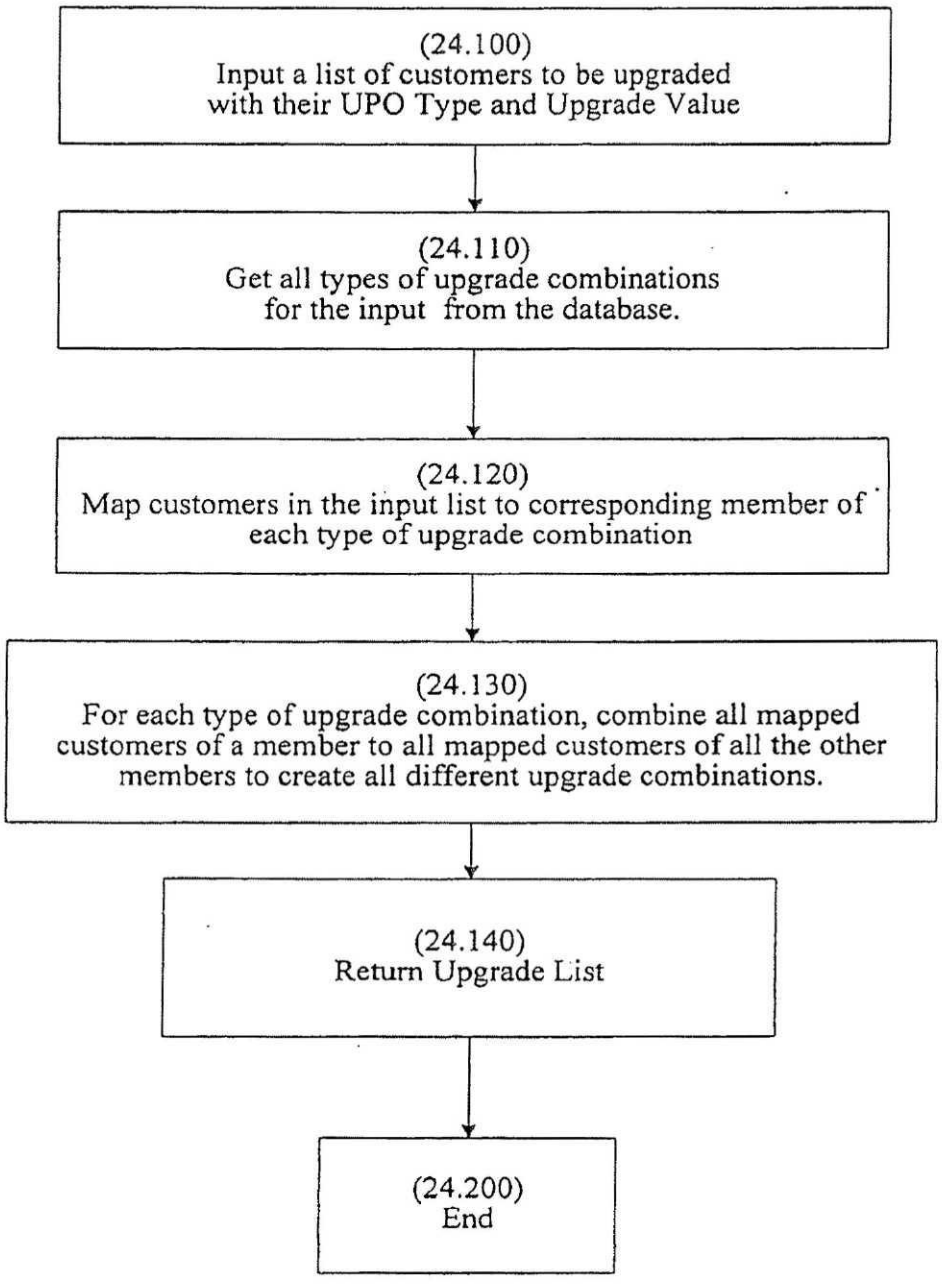

(24.100)
Input a list of customers to be upgraded with their UPO Type and Upgrade Value
(24.110)
Get all types of upgrade combinations for the input from the database.
(24.120)
Map customers in the input list to corresponding member of each type of upgrade combination
(24.130)
For each type of upgrade combination, combine all mapped customers of a member to all mapped customers of all the other members to create all different upgrade combinations.
(24.140)
Return Upgrade List
(24.200)
End

Fig. 25

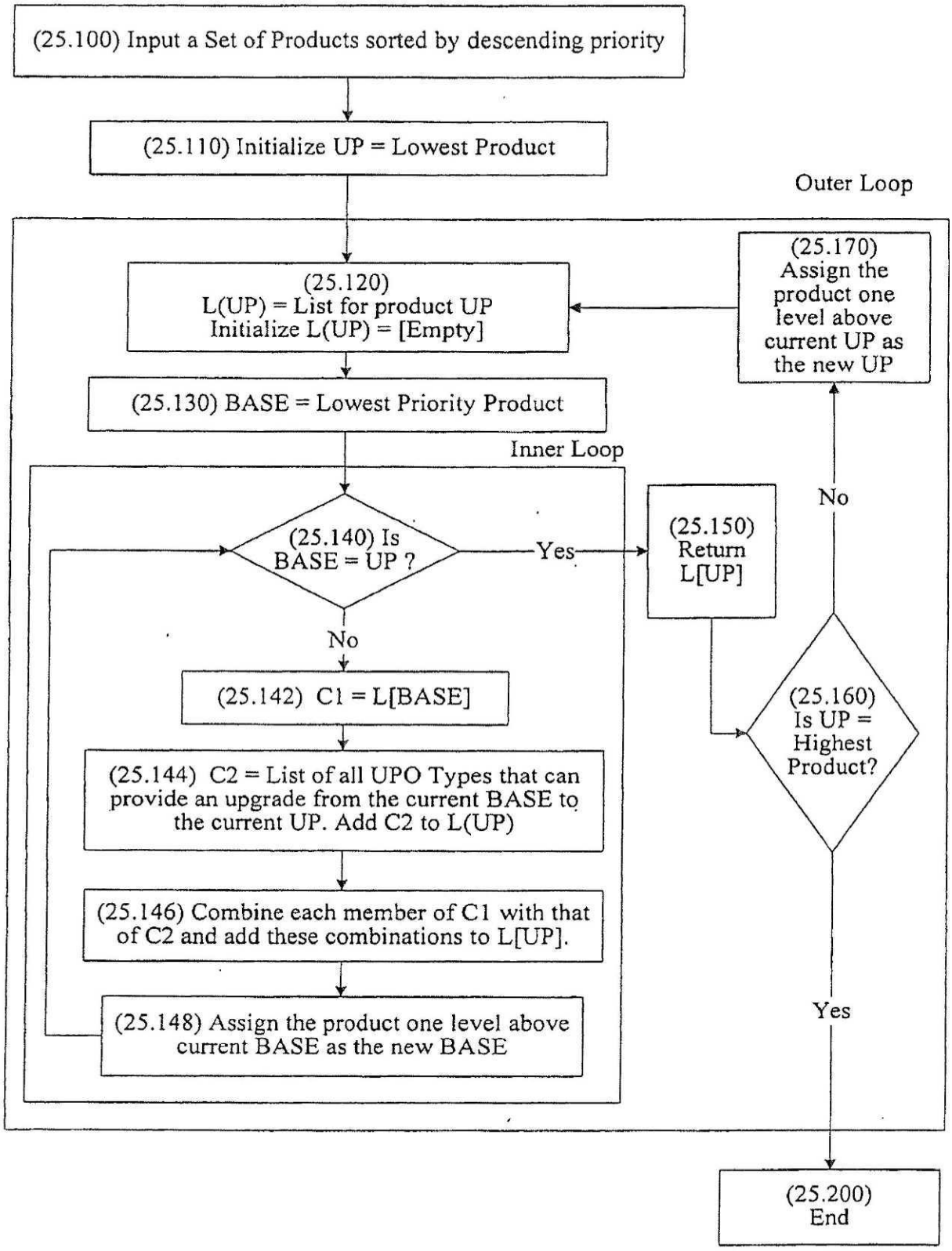

(25.100) Input a Set of Products sorted by descending priority
(25.110) Initialize UP = Lowest Product
Outer Loop
(25.120)
L(UP) = List for product UP
Initialize L(UP) = [Empty]
(25.170)
Assign the product one level above current UP as the new UP
(25.130) BASE = Lowest Priority Product
Inner Loop
(25.140) Is BASE = UP ?
Yes
(25.150)
Return L[UP]
No
No
(25.142) C1 = L[BASE]
(25.160)
Is UP = Highest Product?
(25.144) C2 = List of all UPO Types that can provide an upgrade from the current BASE to the current UP. Add C2 to L(UP)
(25.146) Combine each member of C1 with that of C2 and add these combinations to L[UP].
(25.148) Assign the product one level above current BASE as the new BASE
Yes
(25.200)
End

Fig. 26

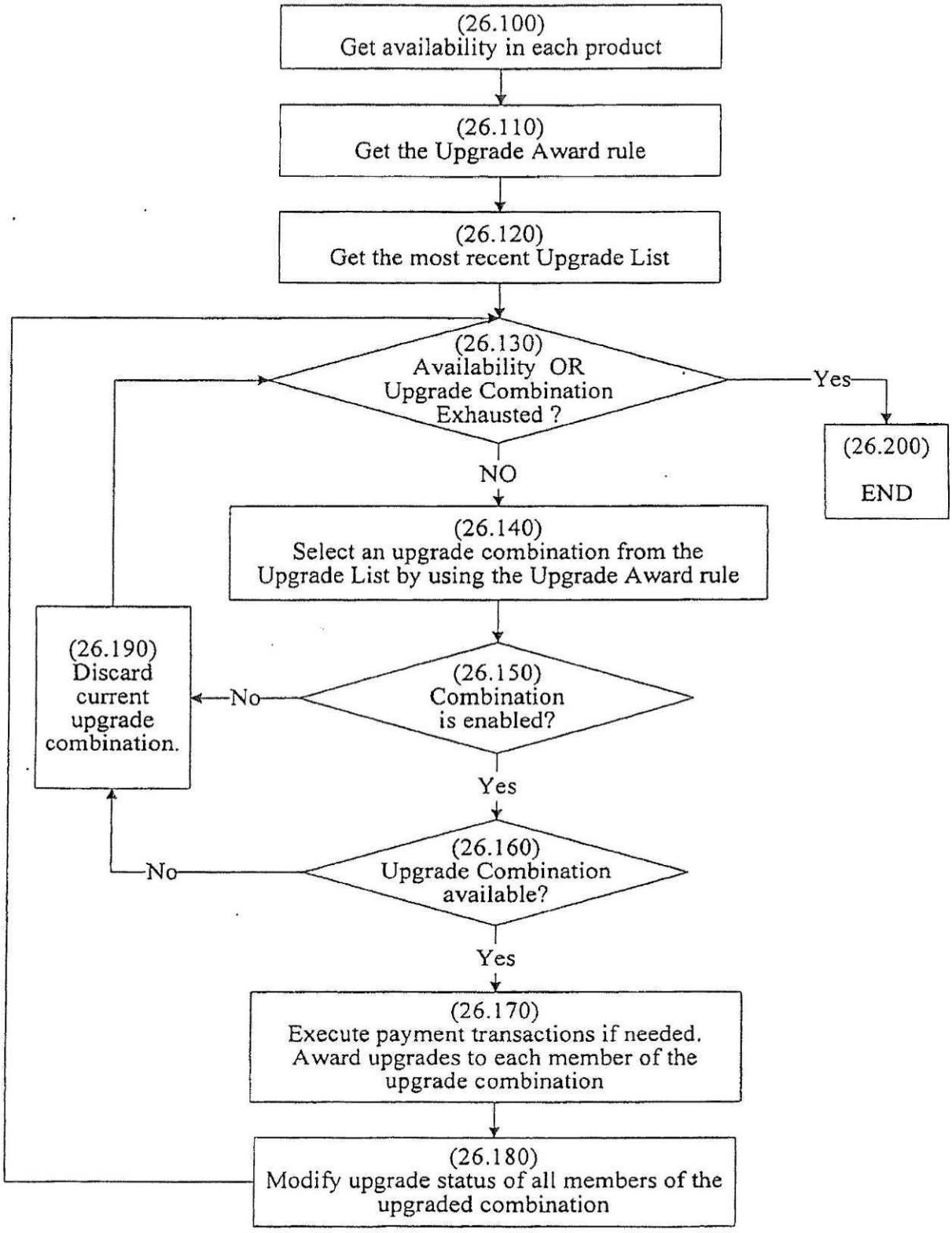

(26.100)
Get availability in each product
(26.110)
Get the Upgrade Award rule
(26.120)
Get the most recent Upgrade List
(26.130)
Availability OR
Upgrade Combination
Exhausted ?
Yes
(26.200)
END
NO
(26.140)
Select an upgrade combination from the
Upgrade List by using the Upgrade Award rule
(26.150)
Combination
is enabled?
No
(26.190)
Discard
current
upgrade
combination.
Yes
(26.160)
Upgrade Combination
available?
No
Yes
(26.170)
Execute payment transactions if needed.
Award upgrades to each member of the
upgrade combination
(26.180)
Modify upgrade status of all members of the
upgraded combination

Fig. 27

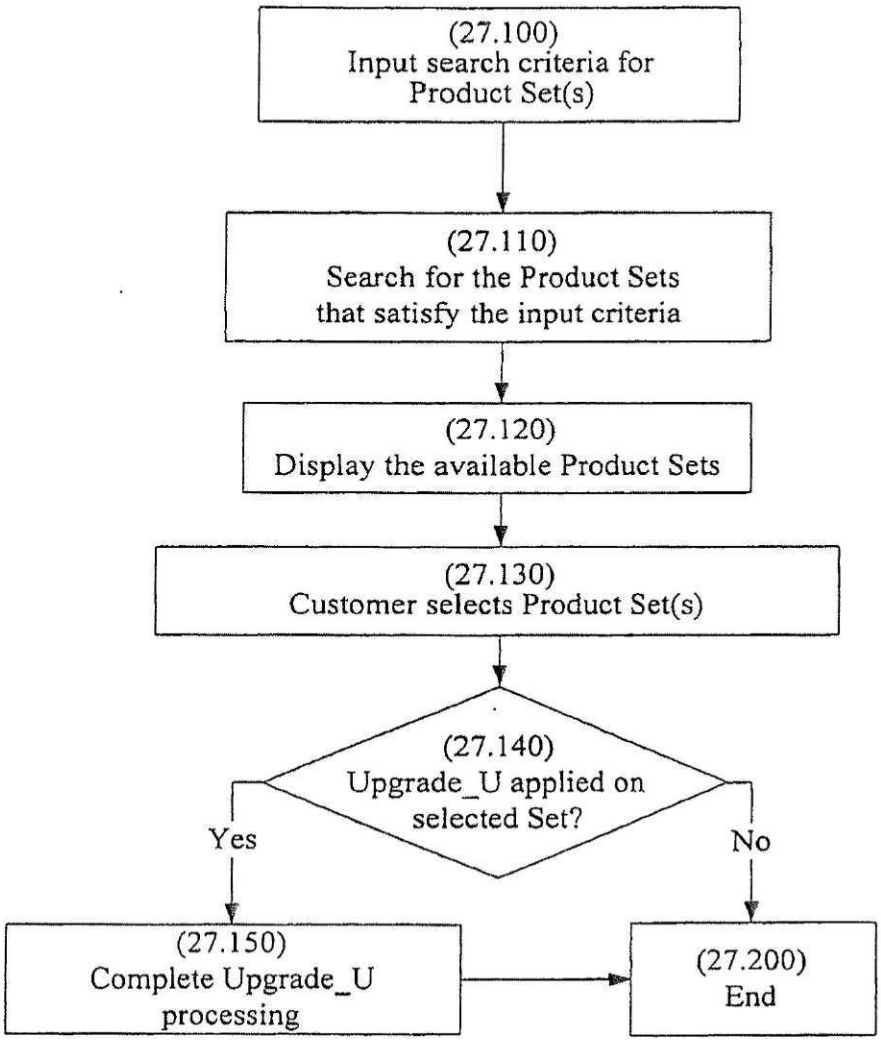

(27.100)
Input search criteria for Product Set(s)
(27.110)
Search for the Product Sets that satisfy the input criteria
(27.120)
Display the available Product Sets
(27.130)
Customer selects Product Set(s)
(27.140)
Upgrade_U applied on selected Set?
Yes
No
(27.150)
Complete Upgrade_U processing
(27.200)
End

Fig. 28

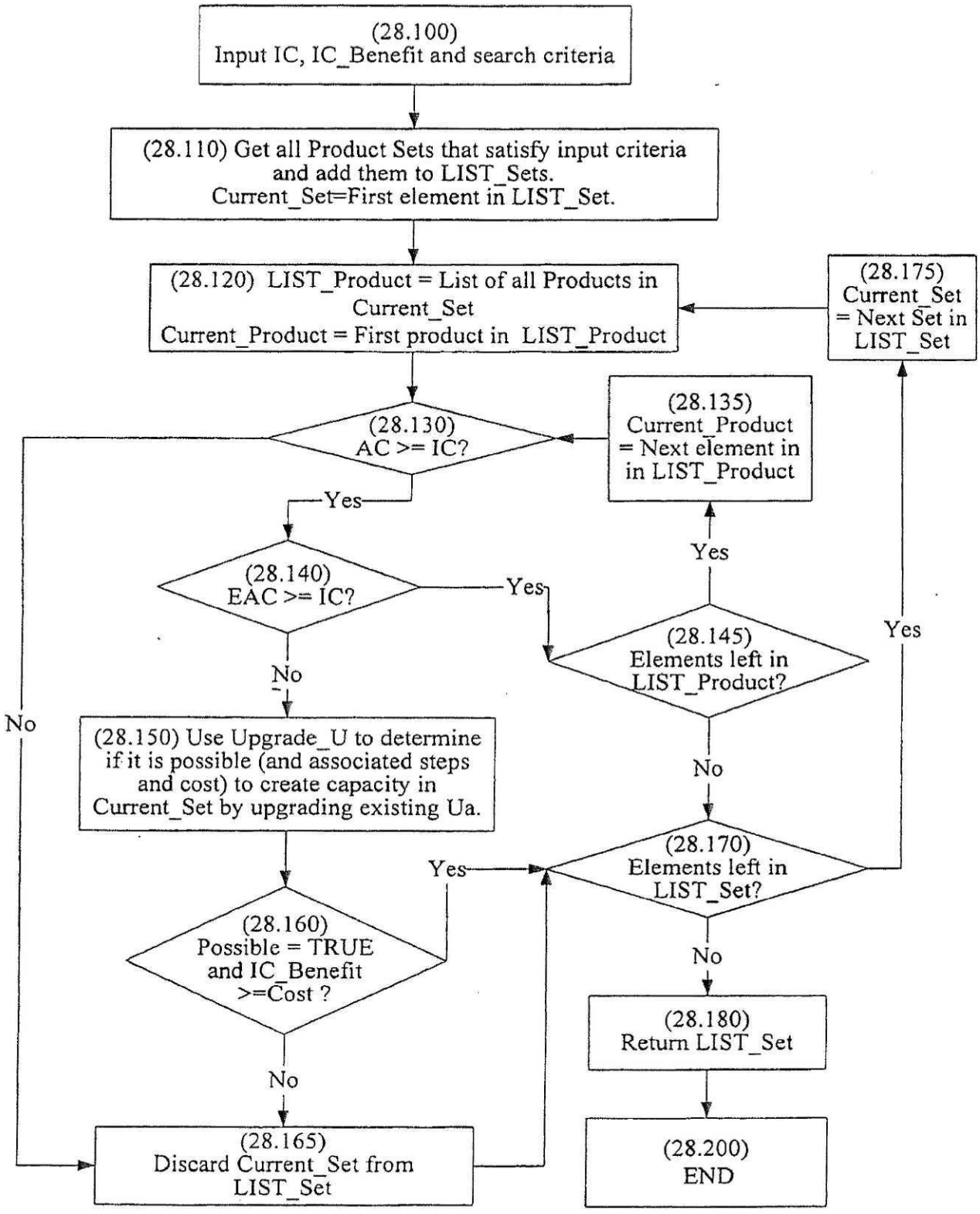

(28.100)
Input IC, IC_Benefit and search criteria
(28.110) Get all Product Sets that satisfy input criteria and add them to LIST_Sets.
Current_Set=First element in LIST_Set.
(28.120) LIST_Product = List of all Products in Current_Set
Current_Product = First product in LIST_Product
(28.175) Current_Set = Next Set in LIST_Set
(28.130) AC >= IC?
(28.135) Current_Product = Next element in in LIST_Product
Yes
(28.140) EAC >= IC?
Yes
(28.145) Elements left in LIST_Product?
Yes
No
No
(28.150) Use Upgrade_U to determine if it is possible (and associated steps and cost) to create capacity in Current_Set by upgrading existing Ua.
No
(28.170) Elements left in LIST_Set?
Yes
(28.160) Possible = TRUE and IC_Benefit >=Cost ?
No
(28.180) Return LIST_Set
No
(28.165) Discard Current_Set from LIST_Set
(28.200) END

Fig. 29

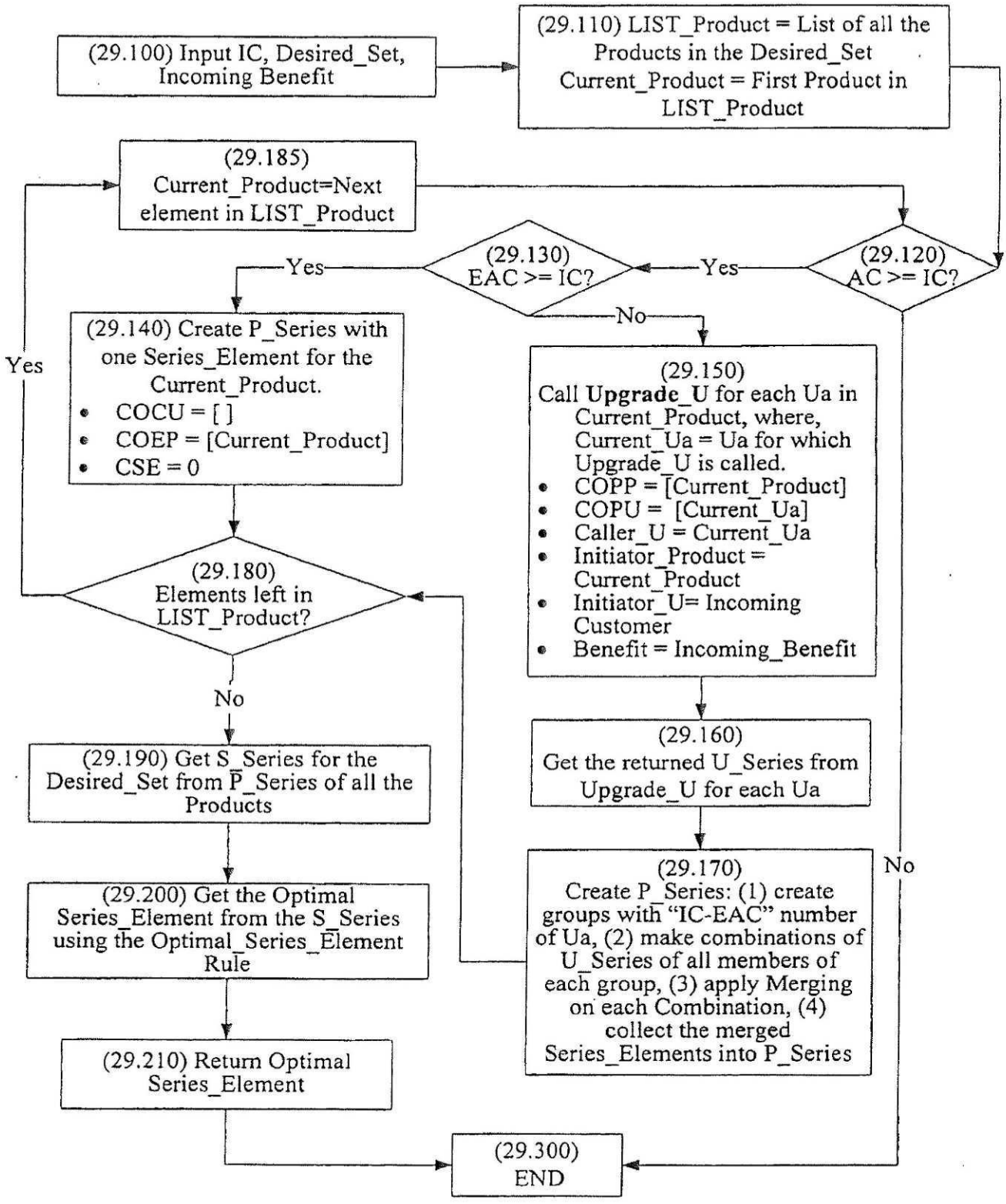

(29.100) Input IC, Desired_Set, Incoming Benefit
(29.110) LIST_Product = List of all the Products in the Desired_Set Current_Product = First Product in LIST_Product
(29.185) Current_Product=Next element in LIST_Product
(29.120) AC >= IC?
Yes
(29.130) EAC >= IC?
Yes
No
Yes
(29.140) Create P_Series with one Series_Element for the Current_Product.
• COCU = [ ]
• COEP = [Current_Product]
• CSE = 0
(29.150) Call Upgrade_U for each Ua in Current_Product, where, Current_Ua = Ua for which Upgrade_U is called.
• COPP = [Current_Product]
• COPU = [Current_Ua]
• Caller_U = Current_Ua
• Initiator_Product = Current_Product
• Initiator_U= Incoming Customer
• Benefit = Incoming_Benefit
(29.180) Elements left in LIST_Product?
No
(29.160) Get the returned U_Series from Upgrade_U for each Ua
(29.190) Get S_Series for the Desired_Set from P_Series of all the Products
(29.170) Create P_Series: (1) create groups with "IC-EAC" number of Ua, (2) make combinations of U_Series of all members of each group, (3) apply Merging on each Combination, (4) collect the merged Series_Elements into P_Series
No
(29.200) Get the Optimal Series_Element from the S_Series using the Optimal_Series_Element Rule
(29.210) Return Optimal Series_Element
(29.300) END

Fig. 30

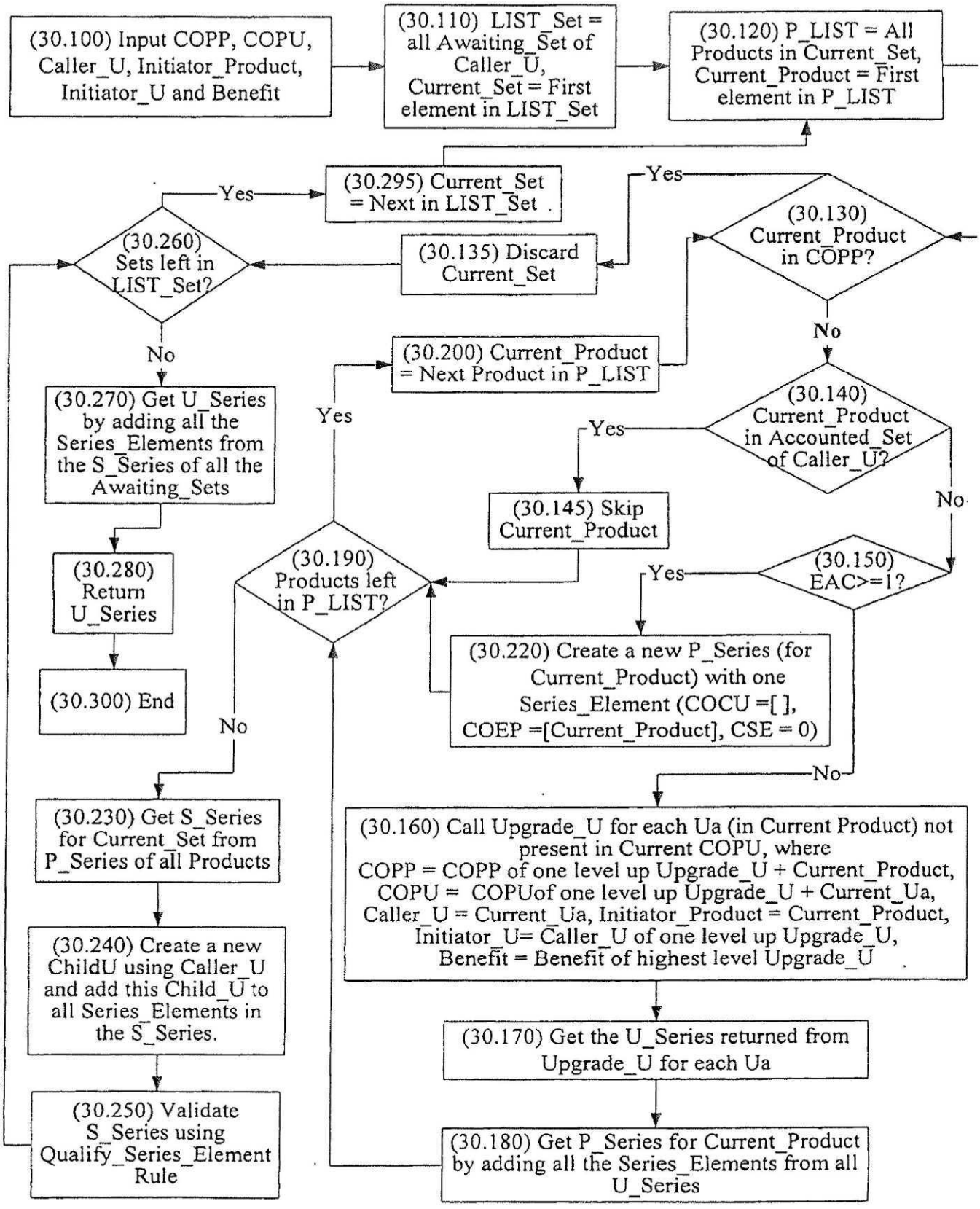

(30.100) Input COPP, COPU, Caller_U, Initiator_Product, Initiator_U and Benefit
(30.110) LIST_Set = all Awaiting_Set of Caller_U, Current_Set = First element in LIST_Set
(30.120) P_LIST = All Products in Current_Set, Current_Product = First element in P_LIST
(30.295) Current_Set = Next in LIST_Set
Yes
(30.260) Sets left in LIST_Set?
(30.135) Discard Current_Set
Yes
(30.130) Current_Product in COPP?
No
(30.200) Current_Product = Next Product in P_LIST
Yes
(30.140) Current_Product in Accounted_Set of Caller_U?
Yes
(30.145) Skip Current_Product
No
(30.150) EAC>=1?
Yes
(30.220) Create a new P_Series (for Current_Product) with one Series_Element (COCU =[ ], COEP =[Current_Product], CSE = 0)
(30.190) Products left in P_LIST?
Yes
No
(30.270) Get U_Series by adding all the Series_Elements from the S_Series of all the Awaiting_Sets
(30.280) Return U_Series
(30.300) End
No
(30.230) Get S_Series for Current_Set from P_Series of all Products
(30.240) Create a new ChildU using Caller_U and add this Child_U to all Series_Elements in the S_Series.
(30.250) Validate S_Series using Qualify_Series_Element Rule
No
(30.160) Call Upgrade_U for each Ua (in Current Product) not present in Current COPU, where COPP = COPP of one level up Upgrade_U + Current_Product, COPU = COPUof one level up Upgrade_U + Current_Ua, Caller_U = Current_Ua, Initiator_Product = Current_Product, Initiator_U= Caller_U of one level up Upgrade_U, Benefit = Benefit of highest level Upgrade_U
(30.170) Get the U_Series returned from Upgrade_U for each Ua
(30.180) Get P_Series for Current_Product by adding all the Series_Elements from all U_Series

Fig. 31

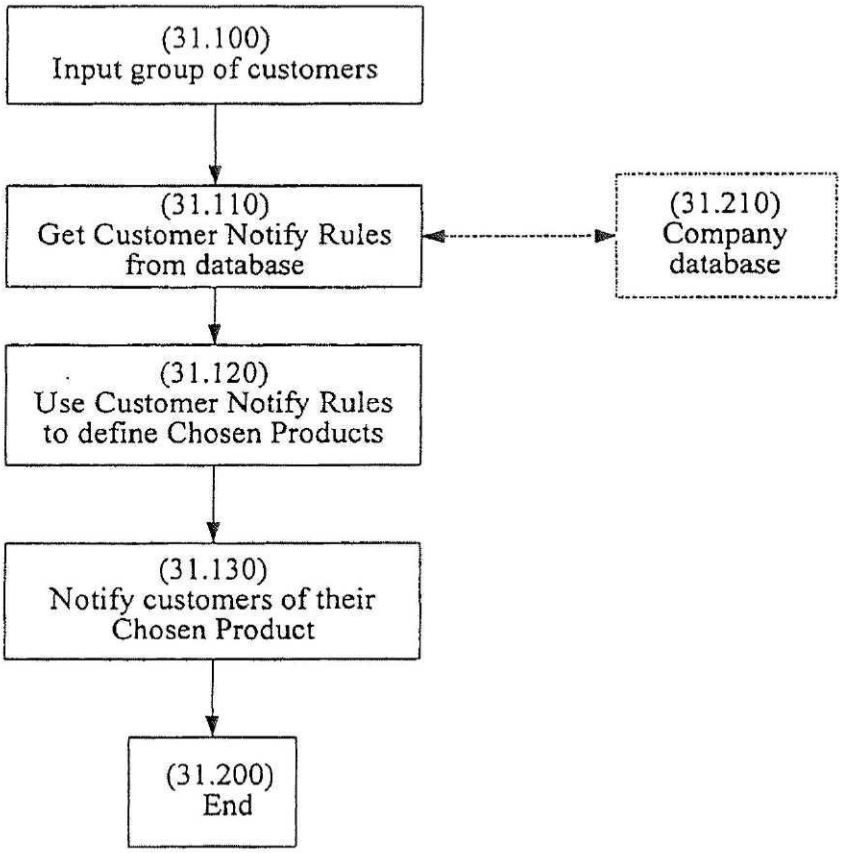

(31.100)
Input group of customers
(31.110)
Get Customer Notify Rules
from database
(31.210)
Company
database
(31.120)
Use Customer Notify Rules
to define Chosen Products
(31.130)
Notify customers of their
Chosen Product
(31.200)
End

Fig. 32

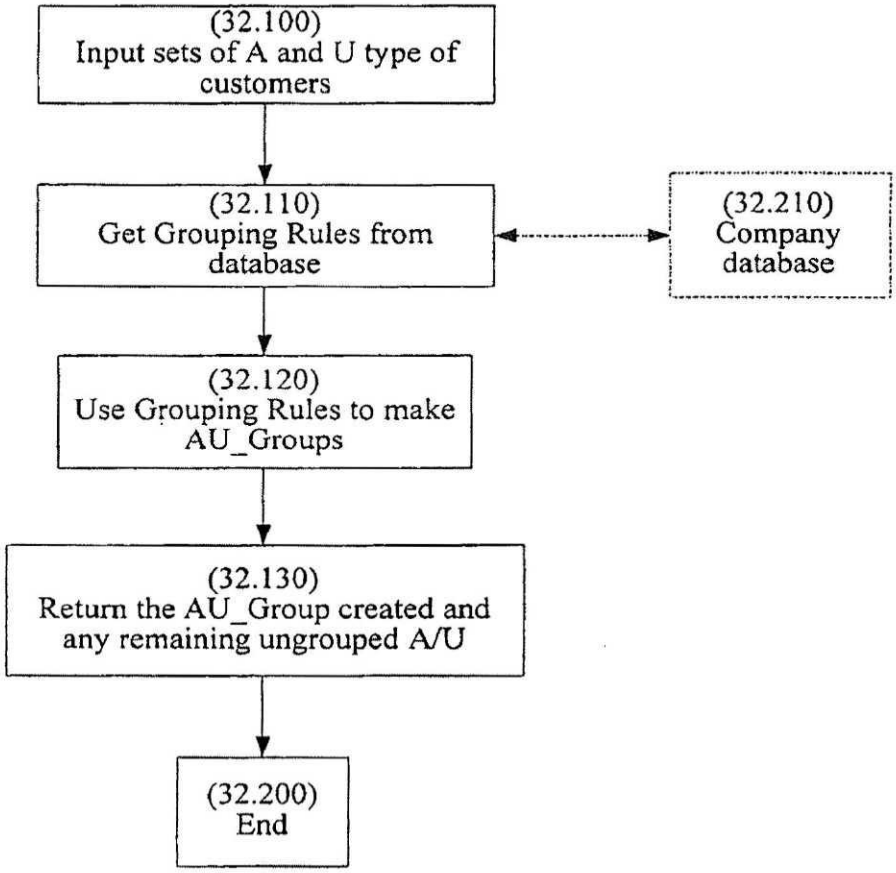

(32.100)
Input sets of A and U type of customers
(32.110)
Get Grouping Rules from database
(32.210)
Company database
(32.120)
Use Grouping Rules to make AU_Groups
(32.130)
Return the AU_Group created and any remaining ungrouped A/U
(32.200)
End

Fig. 33

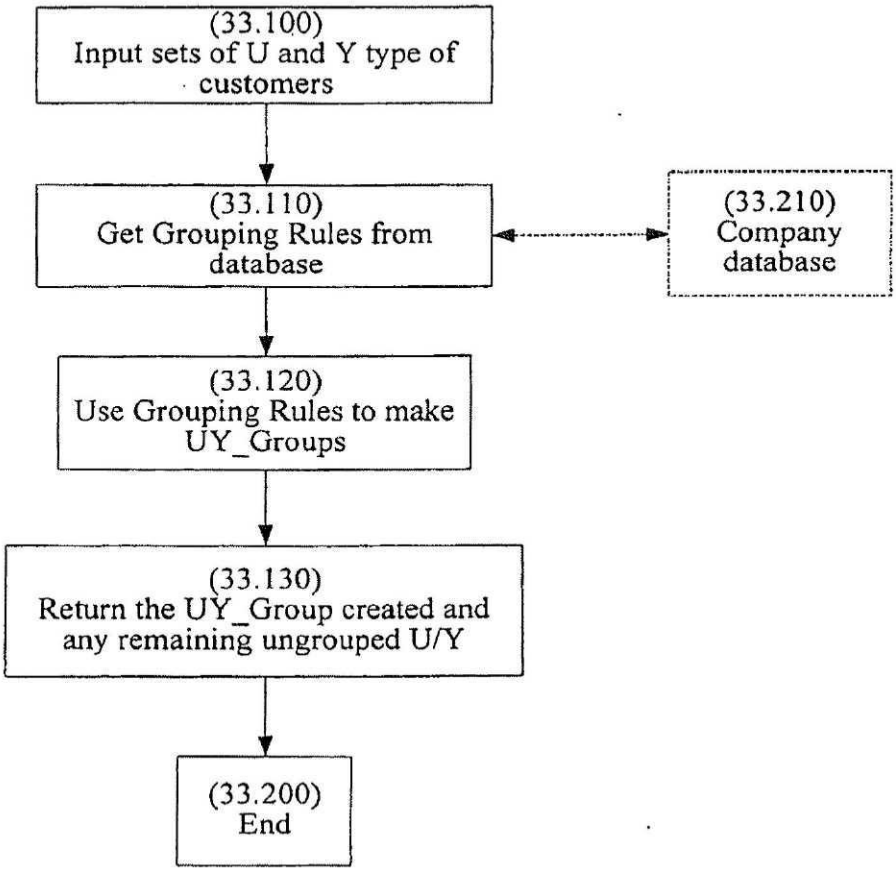

(33.100)
Input sets of U and Y type of customers
(33.110)
Get Grouping Rules from database
(33.210)
Company database
(33.120)
Use Grouping Rules to make UY_Groups
(33.130)
Return the UY_Group created and any remaining ungrouped U/Y
(33.200)
End

Fig. 34

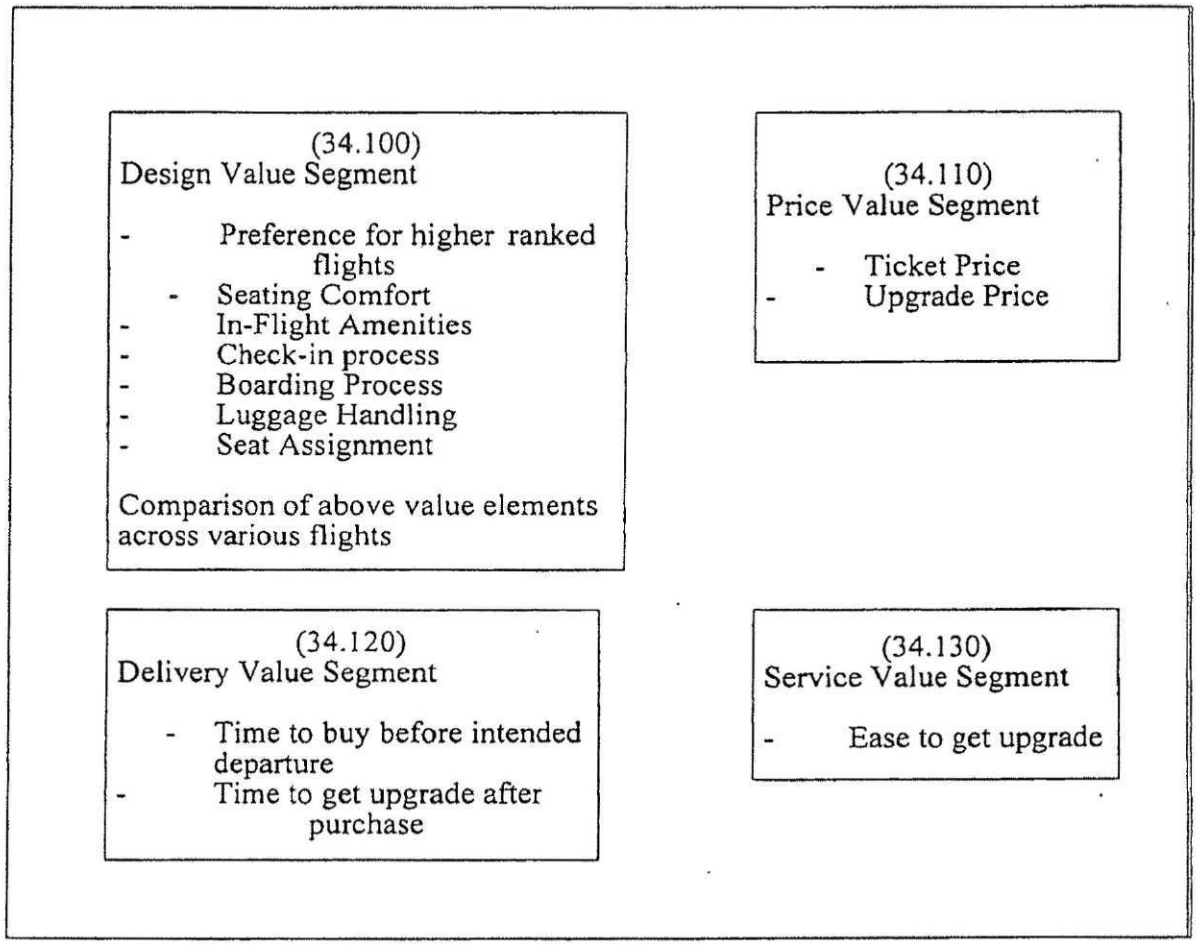

(34.100)
Design Value Segment
- Preference for higher ranked flights
- Seating Comfort
- In-Flight Amenities
- Check-in process
- Boarding Process
- Luggage Handling
- Seat Assignment
Comparison of above value elements across various flights
(34.110)
Price Value Segment
- Ticket Price
- Upgrade Price
(34.120)
Delivery Value Segment
- Time to buy before intended departure
- Time to get upgrade after purchase
(34.130)
Service Value Segment
- Ease to get upgrade

Fig. 35

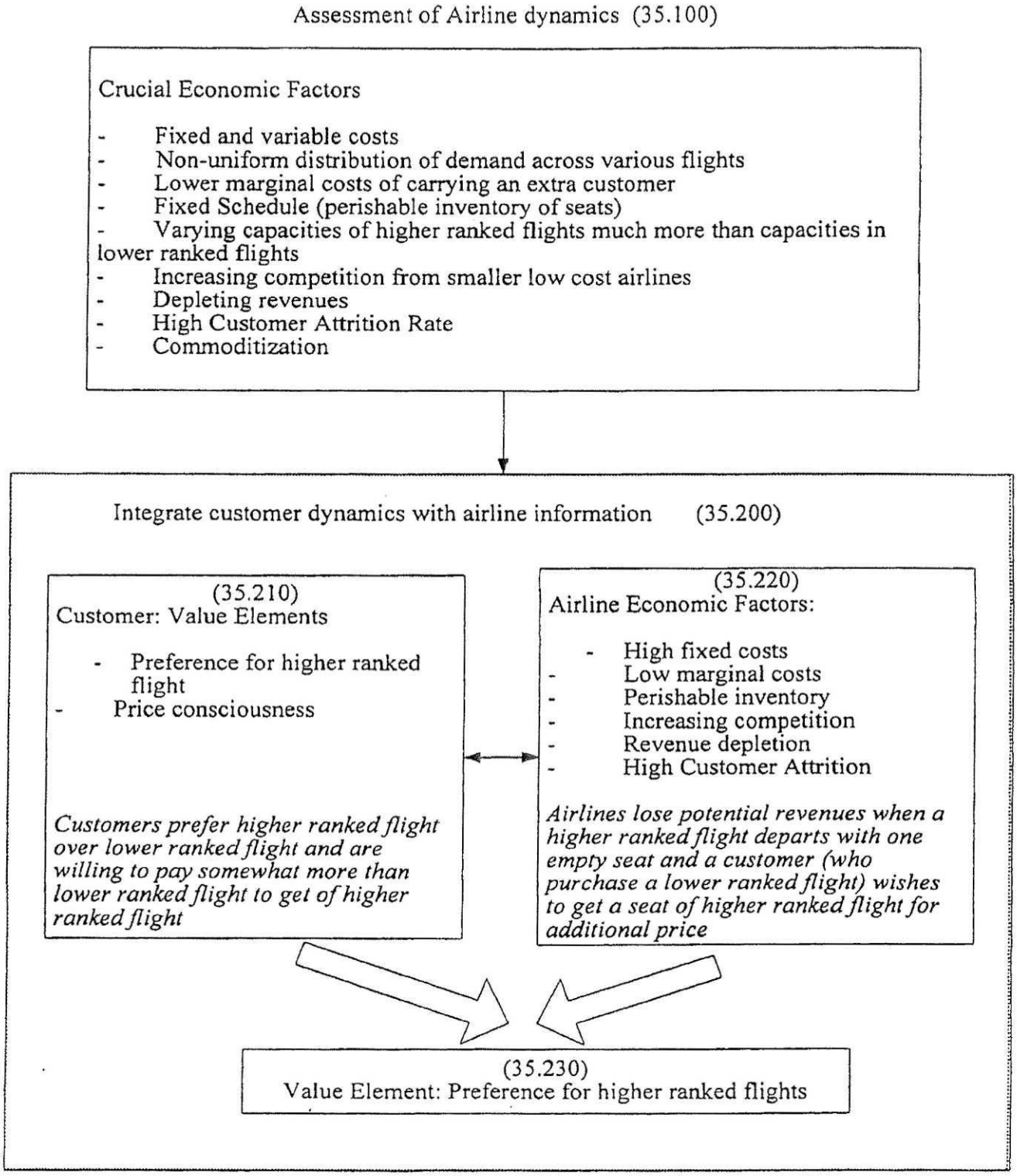

Assessment of Airline dynamics (35.100)
Crucial Economic Factors
- Fixed and variable costs
- Non-uniform distribution of demand across various flights
- Lower marginal costs of carrying an extra customer
- Fixed Schedule (perishable inventory of seats)
- Varying capacities of higher ranked flights much more than capacities in lower ranked flights
- Increasing competition from smaller low cost airlines
- Depleting revenues
- High Customer Attrition Rate
- Commoditization
Integrate customer dynamics with airline information (35.200)
(35.210)
Customer: Value Elements
- Preference for higher ranked flight
- Price consciousness
Customers prefer higher ranked flight over lower ranked flight and are willing to pay somewhat more than lower ranked flight to get of higher ranked flight
(35.220)
Airline Economic Factors:
- High fixed costs
- Low marginal costs
- Perishable inventory
- Increasing competition
- Revenue depletion
- High Customer Attrition
Airlines lose potential revenues when a higher ranked flight departs with one empty seat and a customer (who purchase a lower ranked flight) wishes to get a seat of higher ranked flight for additional price
(35.230)
Value Element: Preference for higher ranked flights

Fig. 36

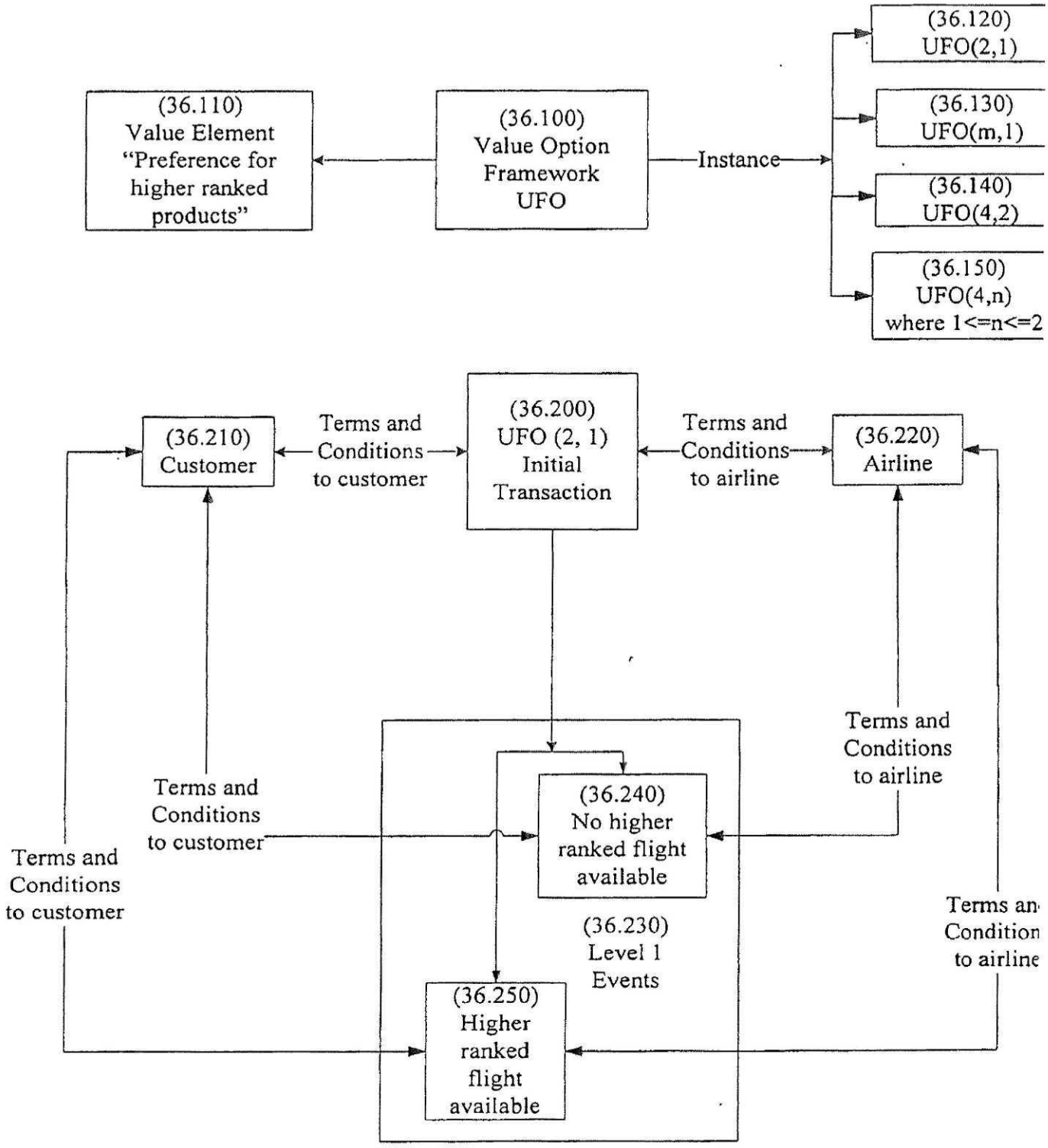

(36.110)
Value Element
"Preference for higher ranked products"
(36.100)
Value Option Framework
UFO
Instance
(36.120)
UFO(2,1)
(36.130)
UFO(m,1)
(36.140)
UFO(4,2)
(36.150)
UFO(4,n)
where 1<=n<=2
(36.210)
Customer
Terms and Conditions to customer
(36.200)
UFO (2, 1)
Initial Transaction
Terms and Conditions to airline
(36.220)
Airline
Terms and Conditions to customer
Terms and Conditions to customer
(36.240)
No higher ranked flight available
Terms and Conditions to airline
(36.230)
Level 1 Events
(36.250)
Higher ranked flight available
Terms an
Condition
to airline

Fig. 37

UFO	Description	Base-Flight	Up Flight
F3-F1	Flight F3 to Flight F1	Flight F3	Flight F1
F3-F2	Flight F3 to Flight F2	Flight F3	Flight F2
F3-F2-F1	Flight F3 to Flight F2 or Flight F1	Flight F3	Flight F2, Flight F1
F2-F1	Flight F2 to Flight F1	Flight F2	Flight F1

Fig. 38

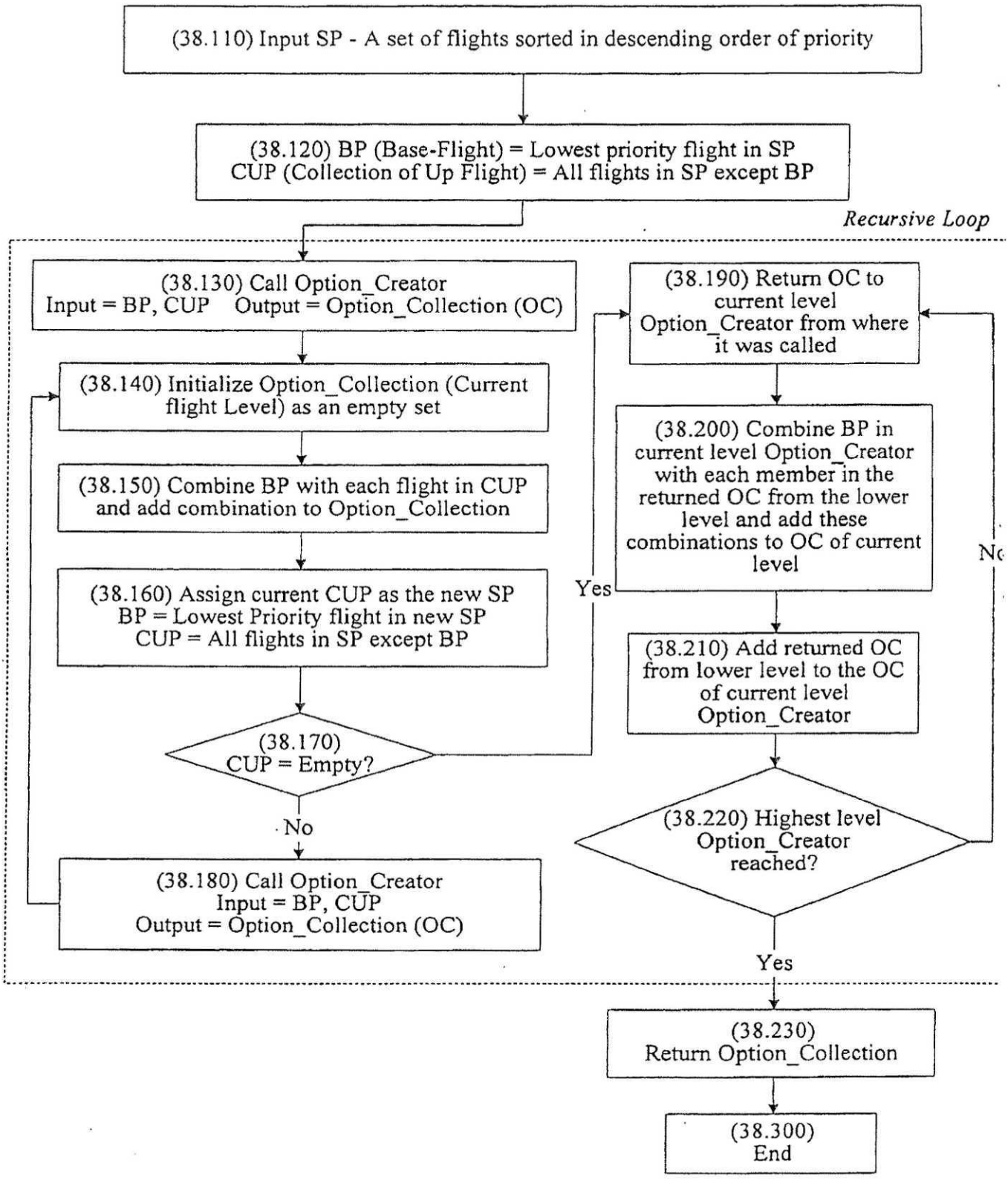

(38.110) Input SP - A set of flights sorted in descending order of priority
(38.120) BP (Base-Flight) = Lowest priority flight in SP
CUP (Collection of Up Flight) = All flights in SP except BP
Recursive Loop
(38.130) Call Option_Creator
Input = BP, CUP Output = Option_Collection (OC)
(38.140) Initialize Option_Collection (Current flight Level) as an empty set
(38.150) Combine BP with each flight in CUP and add combination to Option_Collection
(38.160) Assign current CUP as the new SP
BP = Lowest Priority flight in new SP
CUP = All flights in SP except BP
(38.170)
CUP = Empty?
Yes
No
(38.180) Call Option_Creator
Input = BP, CUP
Output = Option_Collection (OC)
(38.190) Return OC to current level Option_Creator from where it was called
(38.200) Combine BP in current level Option_Creator with each member in the returned OC from the lower level and add these combinations to OC of current level
(38.210) Add returned OC from lower level to the OC of current level Option_Creator
(38.220) Highest level Option_Creator reached?
No
Yes
(38.230)
Return Option_Collection
(38.300)
End

Fig. 39

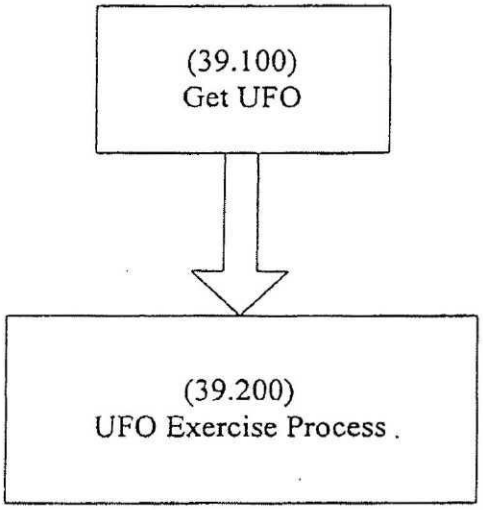


Fig. 40

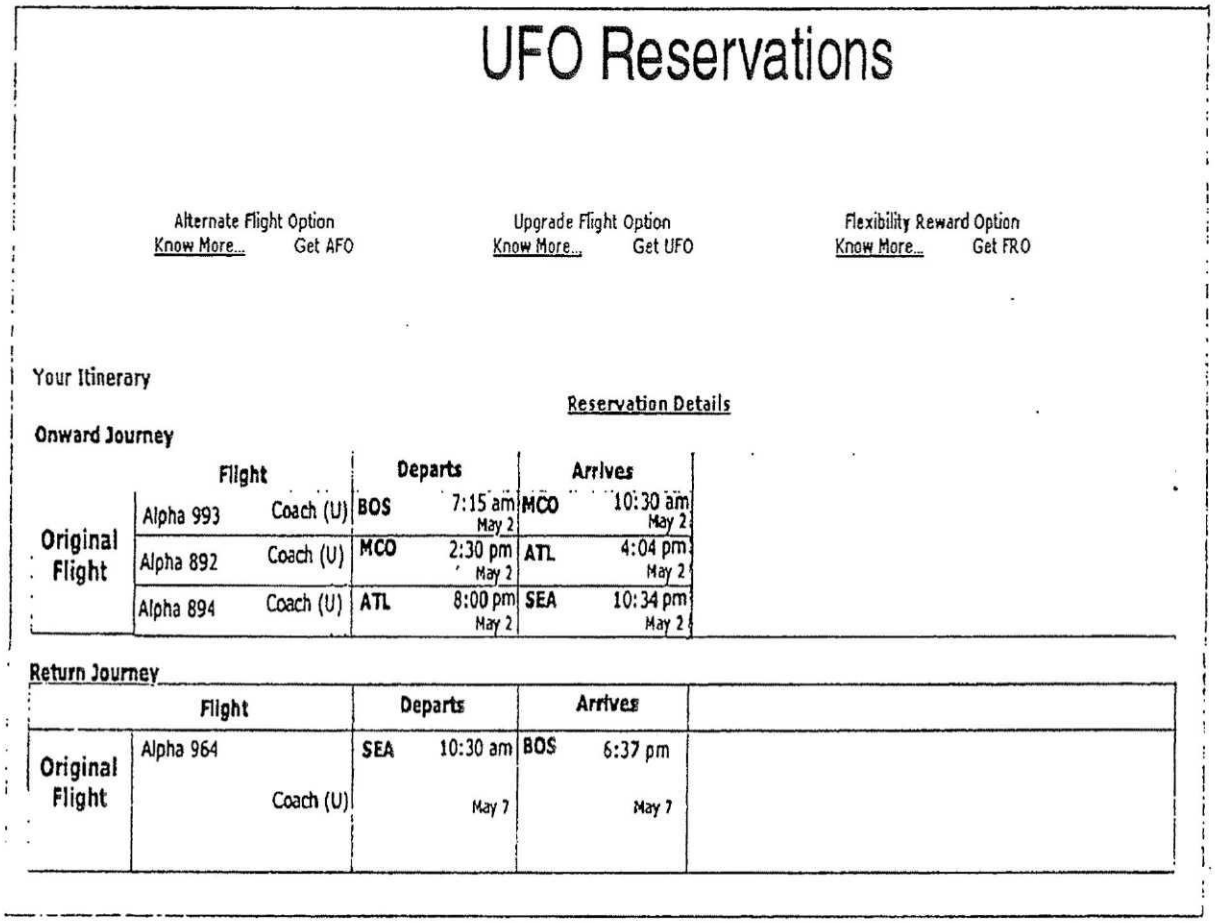

UFO Reservations
Alternate Flight Option
Know More... Get AFO
Upgrade Flight Option
Know More... Get UFO
Flexibility Reward Option
Know More... Get FRO
Your Itinerary
Reservation Details
Onward Journey
Flight
Departs
Arrives
Original Flight
Alpha 993 Coach (U) BOS 7:15 am May 2 MCO 10:30 am May 2
Alpha 892 Coach (U) MCO 2:30 pm May 2 ATL 4:04 pm May 2
Alpha 894 Coach (U) ATL 8:00 pm May 2 SEA 10:34 pm May 2
Return Journey
Flight
Departs
Arrives
Original Flight
Alpha 964 Coach (U) SEA 10:30 am May 7 BOS 6:37 pm May 7

Fig. 41

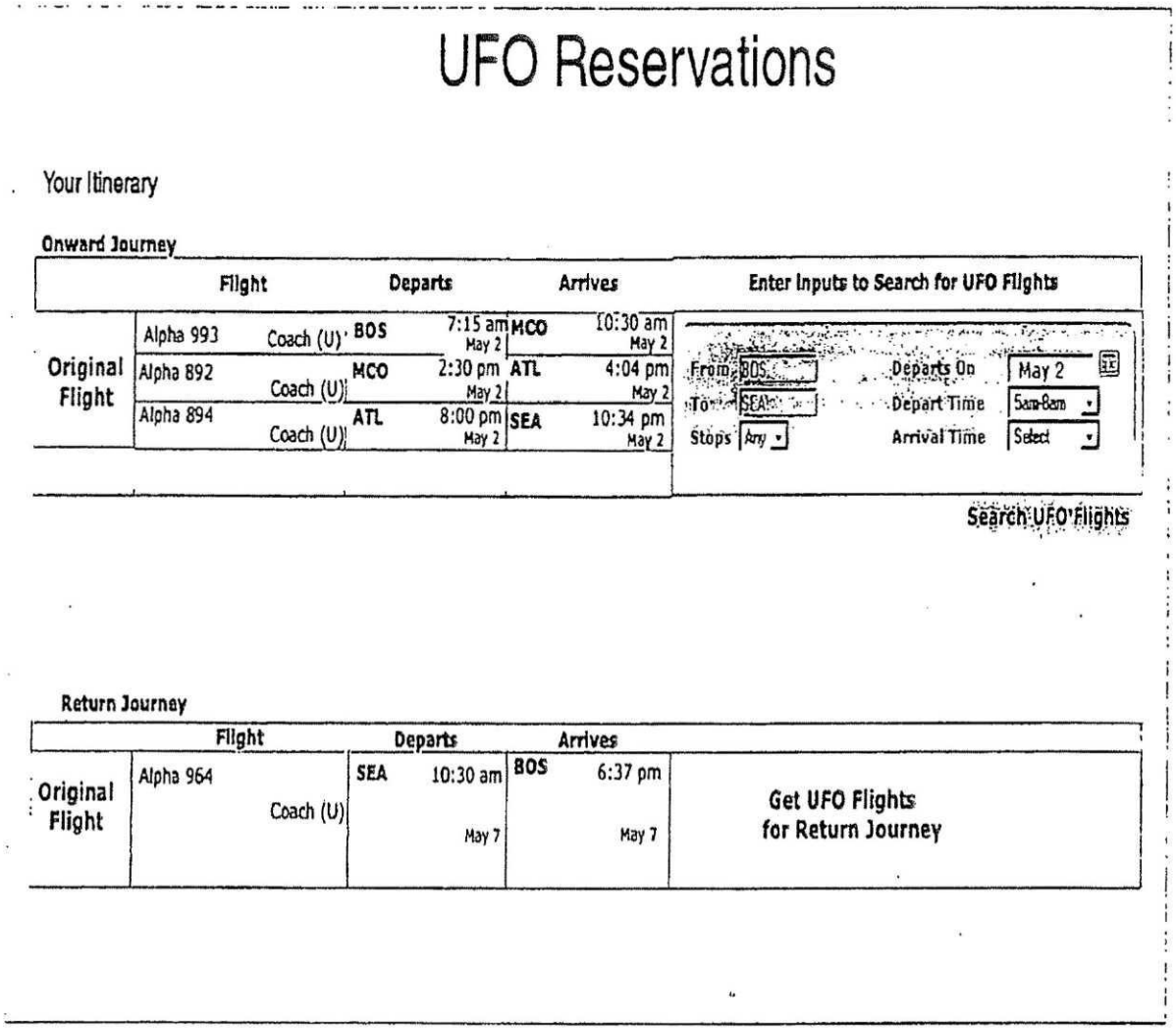

UFO Reservations
Your Itinerary
Onward Journey
Flight
Departs
Arrives
Enter Inputs to Search for UFO Flights
Original Flight
Alpha 993 Coach (U)
BOS 7:15 am May 2
MCO 10:30 am May 2
Alpha 892 Coach (U)
MCO 2:30 pm May 2
ATL 4:04 pm May 2
Alpha 894 Coach (U)
ATL 8:00 pm May 2
SEA 10:34 pm May 2
From BOS
To SEA
Stops Any
Departs On May 2
Depart Time 5am-8am
Arrival Time Select
Search UFO Flights
Return Journey
Flight
Departs
Arrives
Original Flight
Alpha 964 Coach (U)
SEA 10:30 am May 7
BOS 6:37 pm May 7
Get UFO Flights for Return Journey

Fig. 42

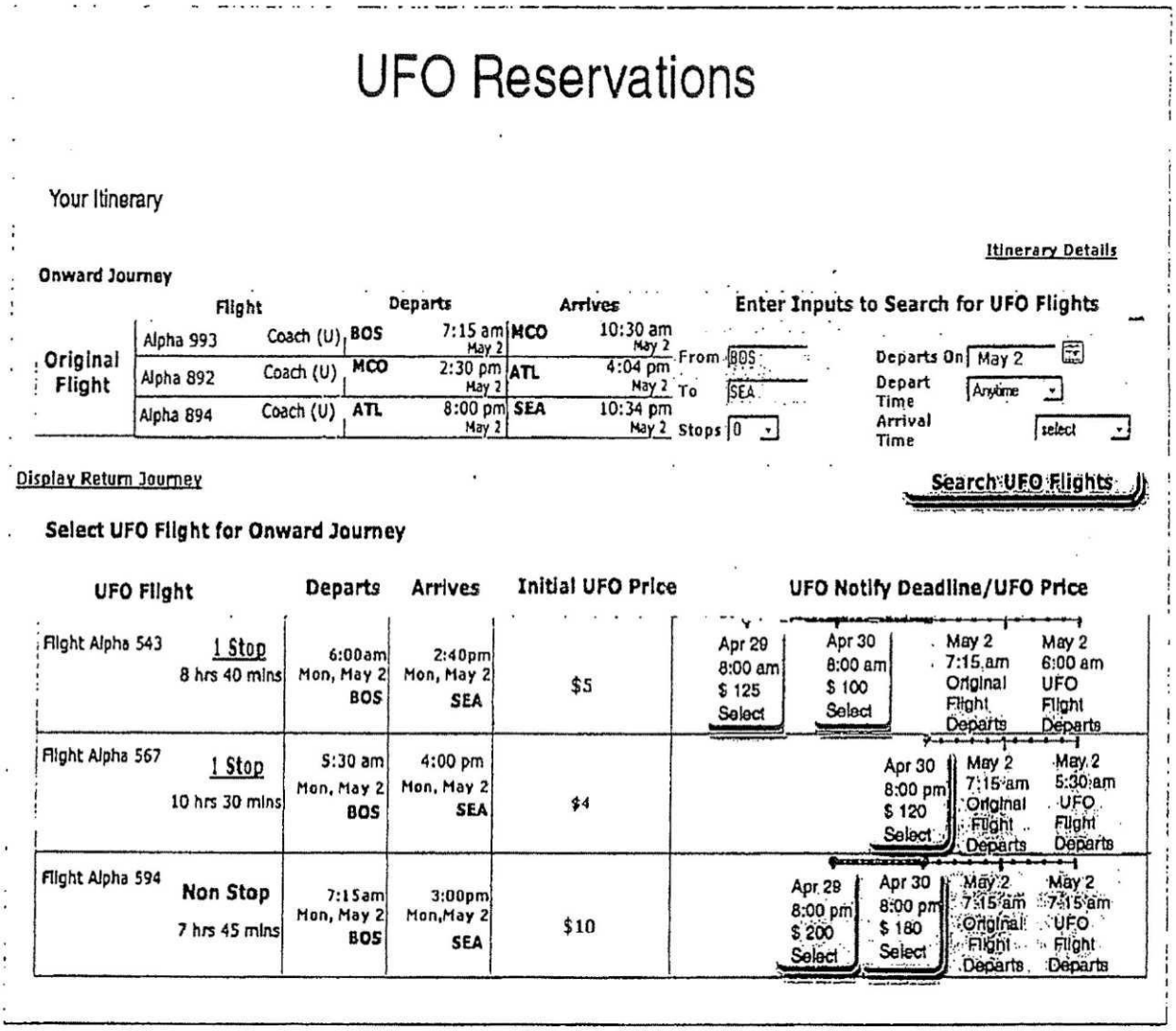

UFO Reservations
Your Itinerary
Itinerary Details
Onward Journey
Flight
Departs
Arrives
Original Flight
Alpha 993 Coach (U) BOS 7:15 am May 2 MCO 10:30 am May 2
Alpha 892 Coach (U) MCO 2:30 pm May 2 ATL 4:04 pm May 2
Alpha 894 Coach (U) ATL 8:00 pm May 2 SEA 10:34 pm May 2
Enter Inputs to Search for UFO Flights
From BOS
To SEA
Stops 0
Departs On May 2
Depart Time Anytime
Arrival Time select
Display Return Journey
Search UFO Flights
Select UFO Flight for Onward Journey
UFO Flight
Departs
Arrives
Initial UFO Price
UFO Notify Deadline/UFO Price
Flight Alpha 543 1 Stop 8 hrs 40 mins 6:00am Mon, May 2 BOS 2:40pm Mon, May 2 SEA $5
Apr 29 8:00 am $ 125 Select
Apr 30 8:00 am $ 100 Select
May 2 7:15 am Original Flight Departs
May 2 6:00 am UFO Flight Departs
Flight Alpha 567 1 Stop 10 hrs 30 mins 5:30 am Mon, May 2 BOS 4:00 pm Mon, May 2 SEA $4
Apr 30 8:00 pm $ 120 Select
May 2 7:15 am Original Flight Departs
May 2 5:30 am UFO Flight Departs
Flight Alpha 594 Non Stop 7 hrs 45 mins 7:15am Mon, May 2 BOS 3:00pm Mon,May 2 SEA $10
Apr 29 8:00 pm $ 200 Select
Apr 30 8:00 pm $ 180 Select
May 2 7:15 am Original Flight Departs
May 2 7:15 am UFO Flight Departs

Fig. 43

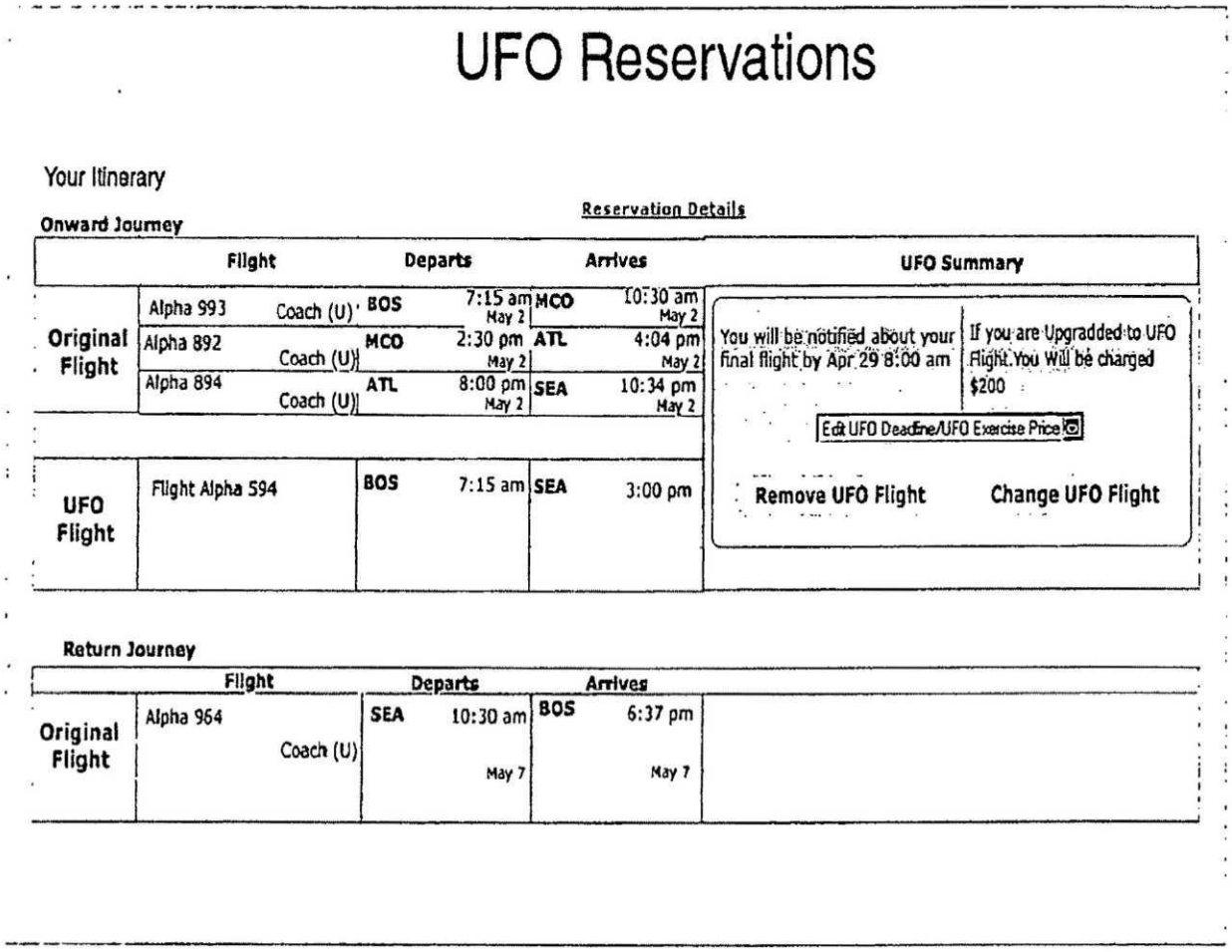

UFO Reservations
Your Itinerary
Reservation Details
Onward Journey
Flight
Departs
Arrives
UFO Summary
Original Flight
Alpha 993 Coach (U)
BOS 7:15 am May 2
MCO 10:30 am May 2
Alpha 892 Coach (U)
MCO 2:30 pm May 2
ATL 4:04 pm May 2
Alpha 894 Coach (U)
ATL 8:00 pm May 2
SEA 10:34 pm May 2
You will be notified about your final flight by Apr 29 8:00 am
If you are Upgradded to UFO Flight. You will be charged $200
Edit UFO Deadline/UFO Exercise Price
UFO Flight
Flight Alpha 594
BOS 7:15 am
SEA 3:00 pm
Remove UFO Flight
Change UFO Flight
Return Journey
Flight
Departs
Arrives
Original Flight
Alpha 964 Coach (U)
SEA 10:30 am May 7
BOS 6:37 pm May 7

Fig. 44

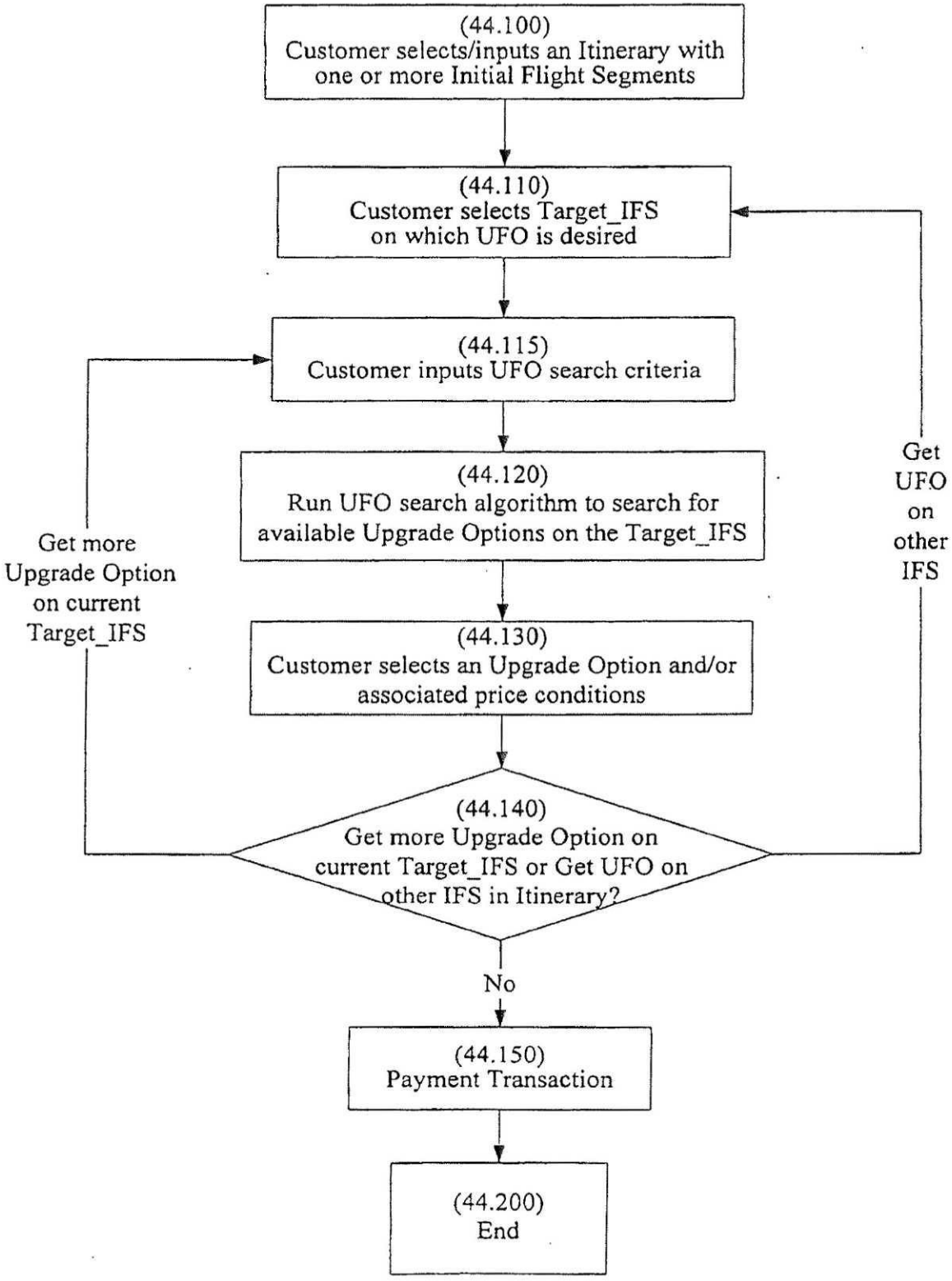

(44.100)
Customer selects/inputs an Itinerary with one or more Initial Flight Segments
(44.110)
Customer selects Target_IFS on which UFO is desired
(44.115)
Customer inputs UFO search criteria
(44.120)
Run UFO search algorithm to search for available Upgrade Options on the Target_IFS
(44.130)
Customer selects an Upgrade Option and/or associated price conditions
(44.140)
Get more Upgrade Option on current Target_IFS or Get UFO on other IFS in Itinerary?
Get more Upgrade Option on current Target_IFS
Get UFO on other IFS
No
(44.150)
Payment Transaction
(44.200)
End

Fig. 45

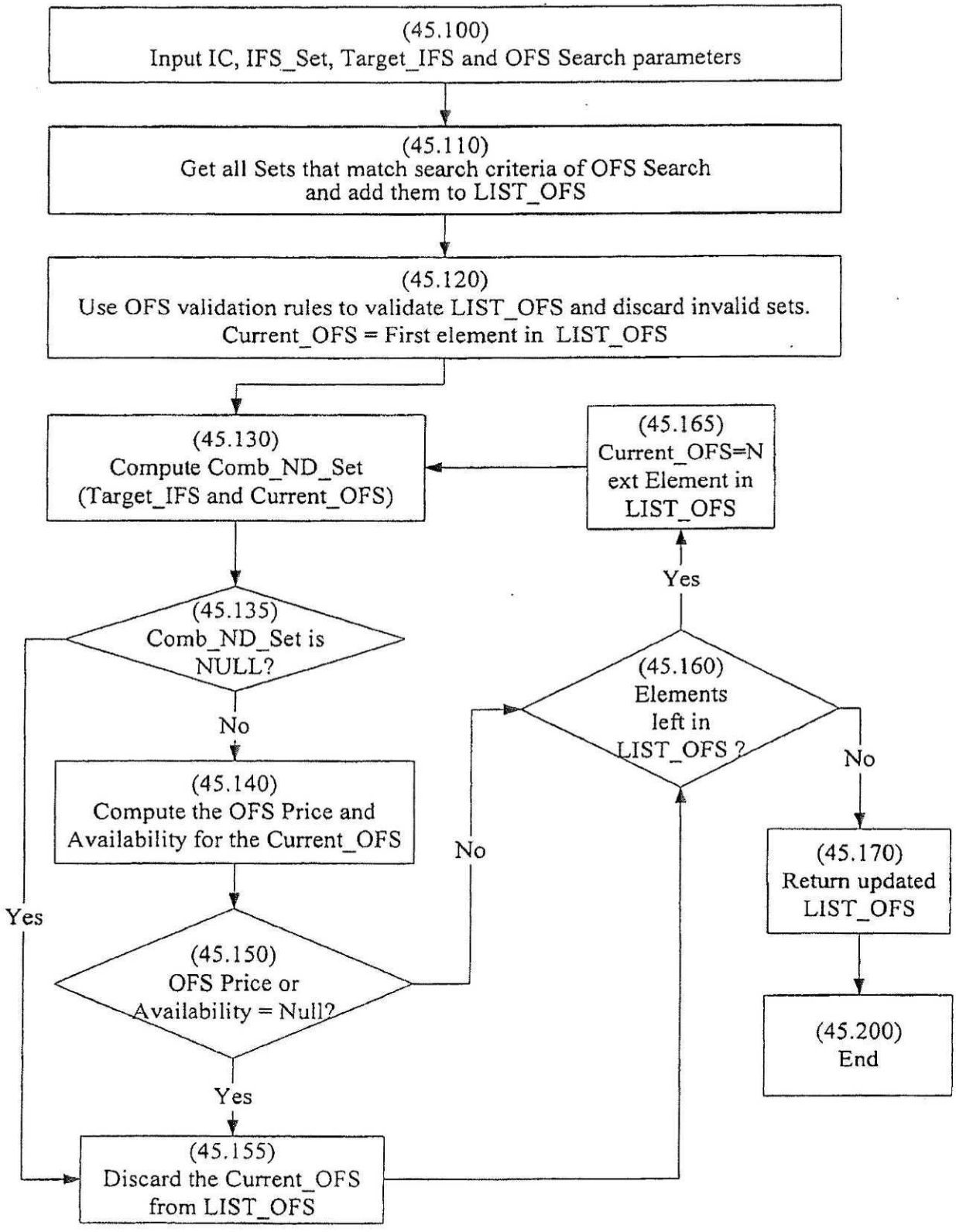

(45.100)
Input IC, IFS_Set, Target_IFS and OFS Search parameters
(45.110)
Get all Sets that match search criteria of OFS Search and add them to LIST_OFS
(45.120)
Use OFS validation rules to validate LIST_OFS and discard invalid sets.
Current_OFS = First element in LIST_OFS
(45.130)
Compute Comb_ND_Set (Target_IFS and Current_OFS)
(45.165)
Current_OFS=Next Element in LIST_OFS
(45.135)
Comb_ND_Set is NULL?
Yes
No
(45.160)
Elements left in LIST_OFS ?
No
Yes
(45.140)
Compute the OFS Price and Availability for the Current_OFS
No
(45.170)
Return updated LIST_OFS
Yes
(45.150)
OFS Price or Availability = Null?
(45.200)
End
Yes
(45.155)
Discard the Current_OFS from LIST_OFS

Fig. 46

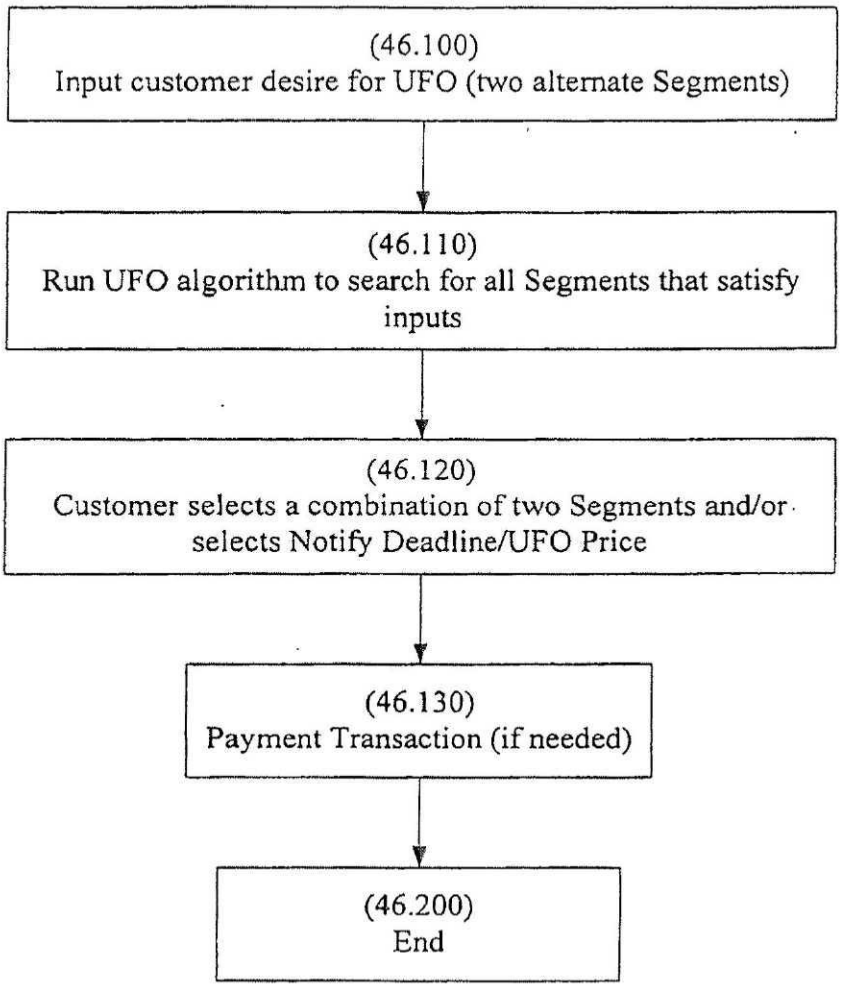

(46.100)
Input customer desire for UFO (two alternate Segments)
(46.110)
Run UFO algorithm to search for all Segments that satisfy inputs
(46.120)
Customer selects a combination of two Segments and/or selects Notify Deadline/UFO Price
(46.130)
Payment Transaction (if needed)
(46.200)
End

Fig. 47

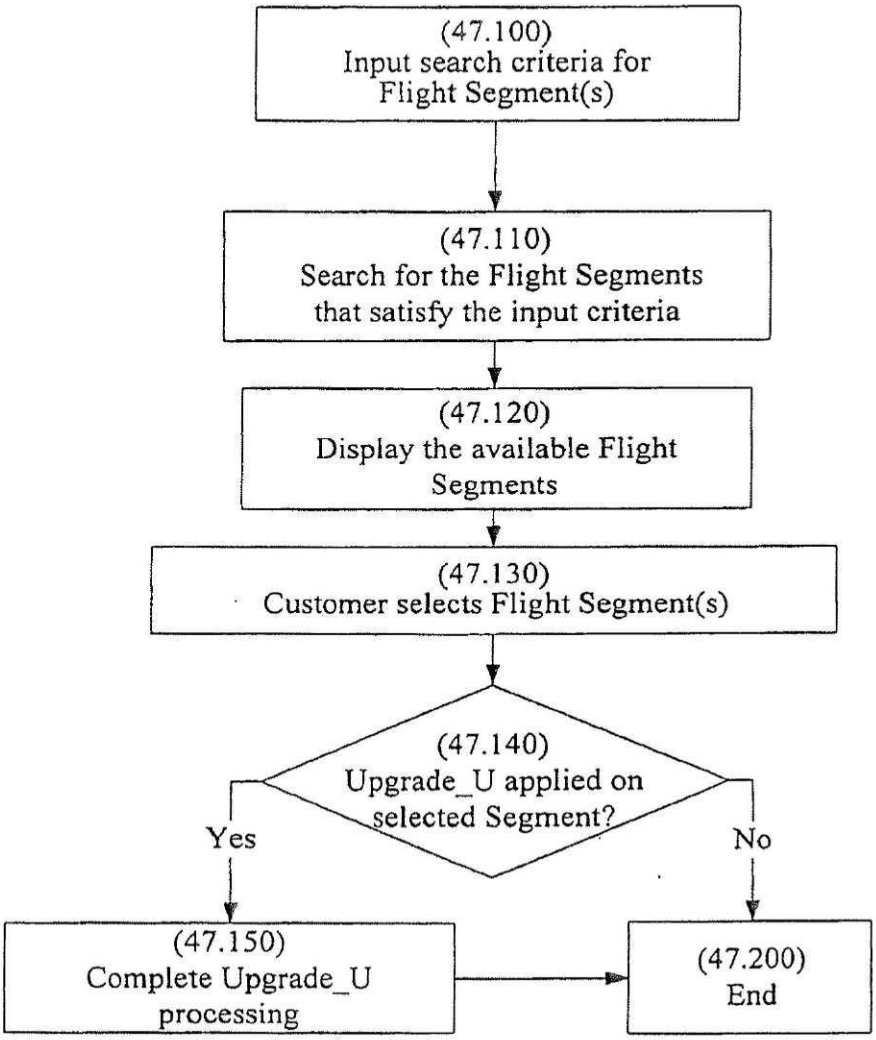

(47.100)
Input search criteria for Flight Segment(s)
(47.110)
Search for the Flight Segments that satisfy the input criteria
(47.120)
Display the available Flight Segments
(47.130)
Customer selects Flight Segment(s)
(47.140)
Upgrade_U applied on selected Segment?
Yes
No
(47.150)
Complete Upgrade_U processing
(47.200)
End

Fig. 48

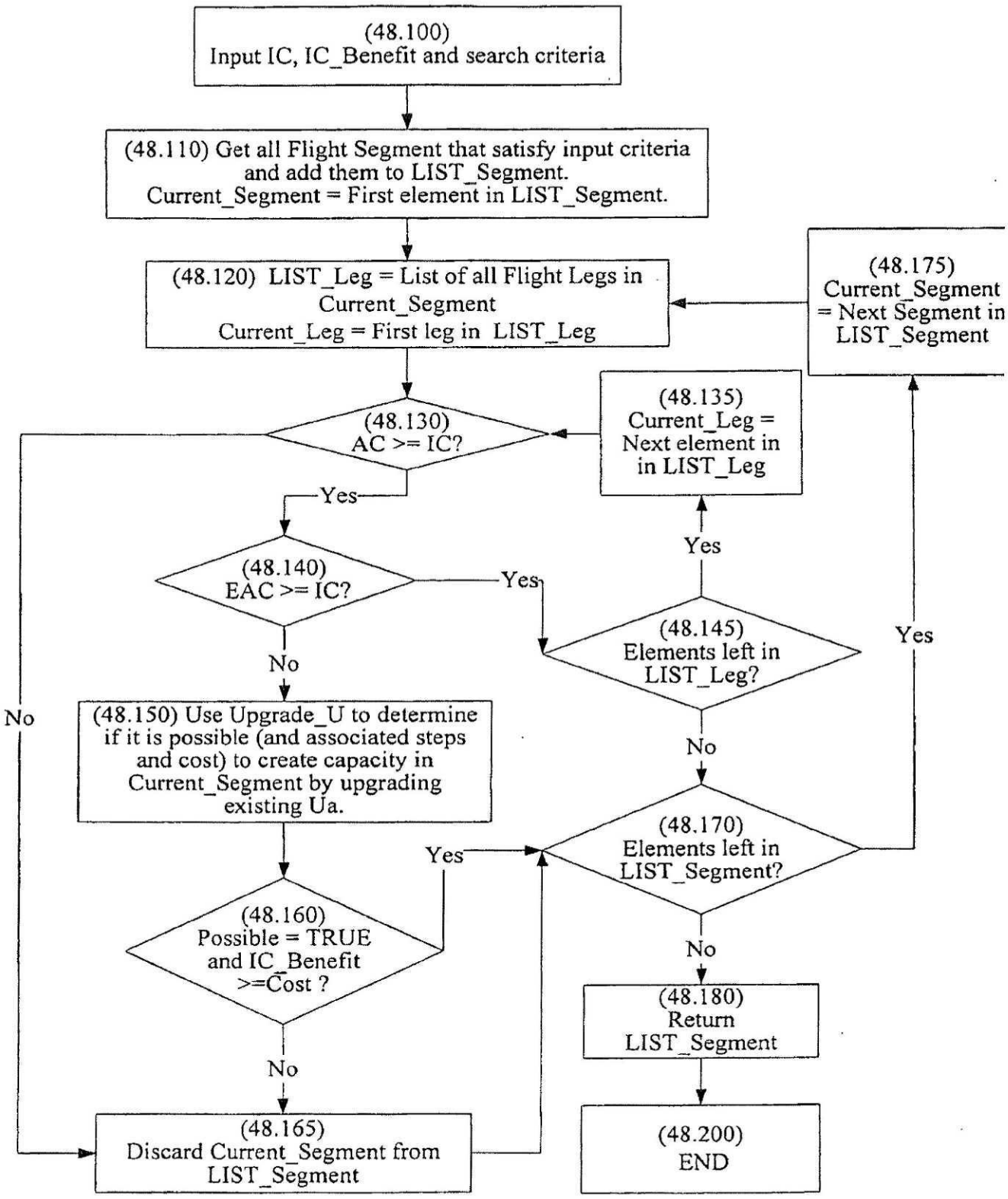

(48.100)
Input IC, IC_Benefit and search criteria
(48.110) Get all Flight Segment that satisfy input criteria and add them to LIST_Segment.
Current_Segment = First element in LIST_Segment.
(48.120) LIST_Leg = List of all Flight Legs in Current_Segment
Current_Leg = First leg in LIST_Leg
(48.175) Current_Segment = Next Segment in LIST_Segment
(48.130) AC >= IC?
(48.135) Current_Leg = Next element in in LIST_Leg
Yes
No
(48.140) EAC >= IC?
Yes
No
(48.145) Elements left in LIST_Leg?
Yes
No
(48.150) Use Upgrade_U to determine if it is possible (and associated steps and cost) to create capacity in Current_Segment by upgrading existing Ua.
(48.170) Elements left in LIST_Segment?
Yes
No
(48.160) Possible = TRUE and IC_Benefit >=Cost ?
Yes
No
(48.165) Discard Current_Segment from LIST_Segment
(48.180) Return LIST_Segment
(48.200) END

Fig. 49

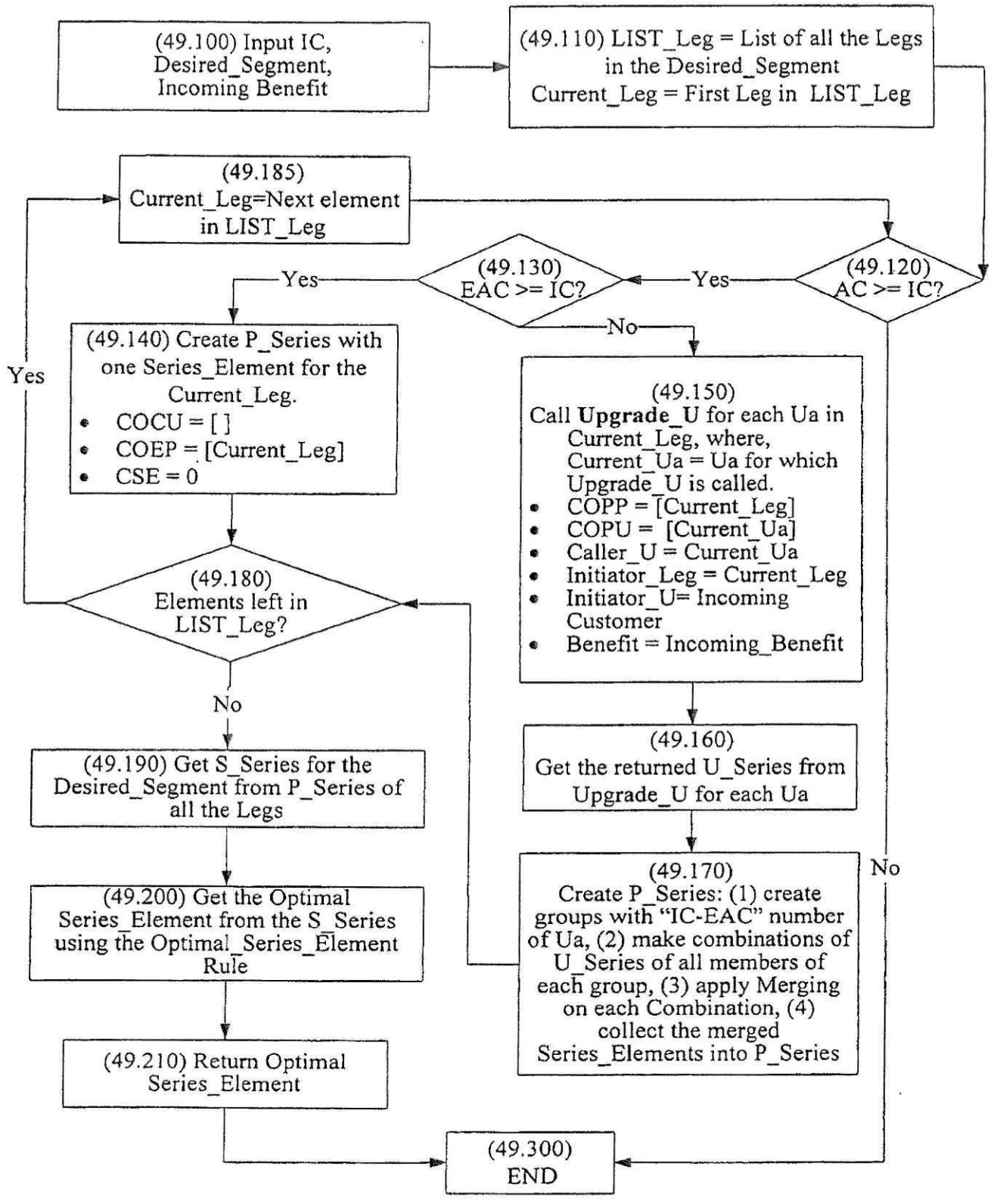

(49.100) Input IC, Desired_Segment, Incoming Benefit
(49.110) LIST_Leg = List of all the Legs in the Desired_Segment Current_Leg = First Leg in LIST_Leg
(49.185) Current_Leg=Next element in LIST_Leg
(49.120) AC >= IC?
(49.130) EAC >= IC?
Yes
Yes
No
(49.140) Create P_Series with one Series_Element for the Current_Leg.
COCU = [ ]
COEP = [Current_Leg]
CSE = 0
(49.150) Call Upgrade_U for each Ua in Current_Leg, where, Current_Ua = Ua for which Upgrade_U is called.
COPP = [Current_Leg]
COPU = [Current_Ua]
Caller_U = Current_Ua
Initiator_Leg = Current_Leg
Initiator_U= Incoming Customer
Benefit = Incoming_Benefit
(49.180) Elements left in LIST_Leg?
Yes
No
(49.160) Get the returned U_Series from Upgrade_U for each Ua
(49.190) Get S_Series for the Desired_Segment from P_Series of all the Legs
(49.170) Create P_Series: (1) create groups with "IC-EAC" number of Ua, (2) make combinations of U_Series of all members of each group, (3) apply Merging on each Combination, (4) collect the merged Series_Elements into P_Series
No
(49.200) Get the Optimal Series_Element from the S_Series using the Optimal_Series_Element Rule
(49.210) Return Optimal Series_Element
(49.300) END

Fig. 50

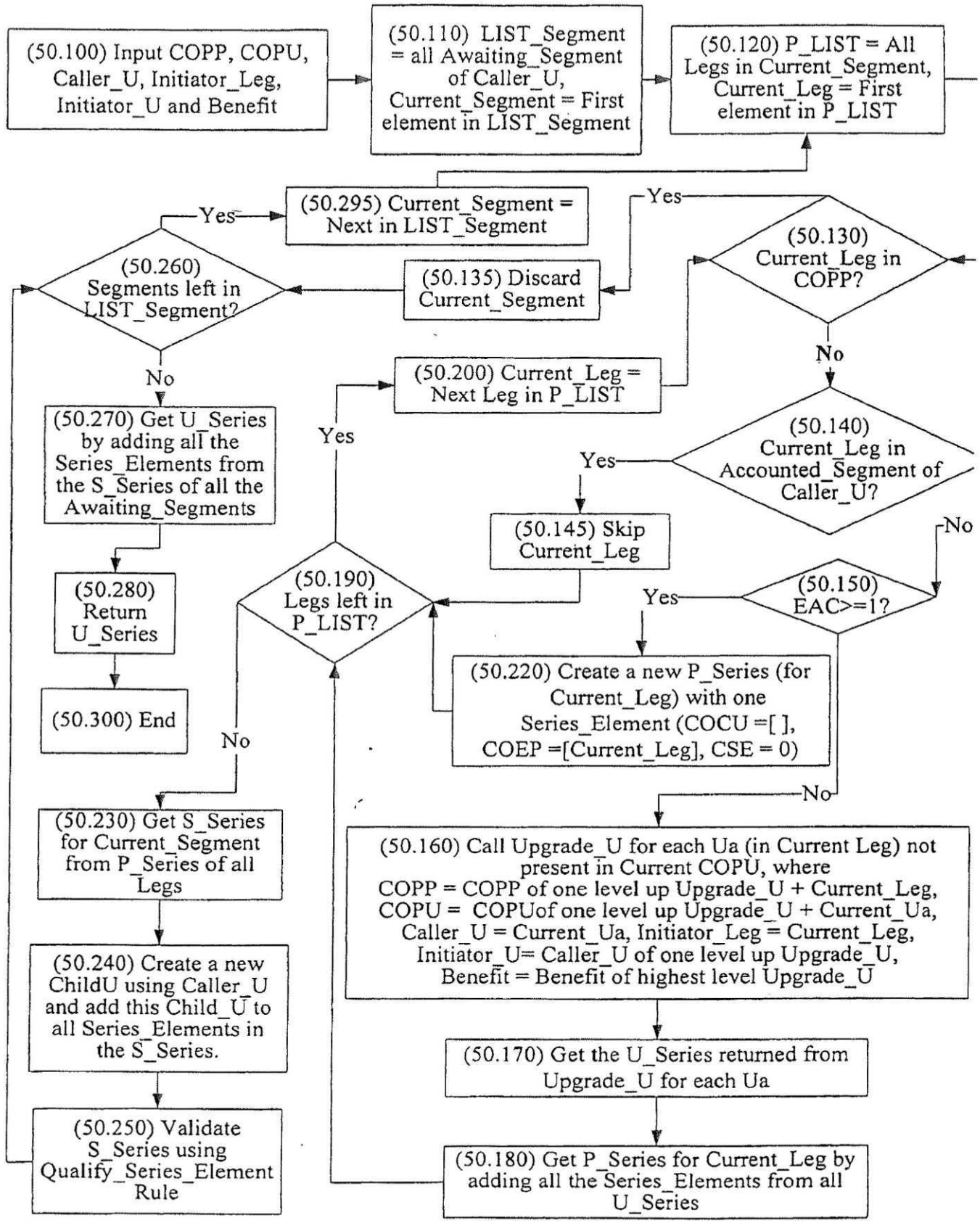

(50.100) Input COPP, COPU, Caller_U, Initiator_Leg, Initiator_U and Benefit
(50.110) LIST_Segment = all Awaiting_Segment of Caller_U, Current_Segment = First element in LIST_Segment
(50.120) P_LIST = All Legs in Current_Segment, Current_Leg = First element in P_LIST
(50.295) Current_Segment = Next in LIST_Segment
Yes
(50.260) Segments left in LIST_Segment?
(50.135) Discard Current_Segment
Yes
(50.130) Current_Leg in COPP?
No
(50.200) Current_Leg = Next Leg in P_LIST
Yes
(50.140) Current_Leg in Accounted_Segment of Caller_U?
Yes
(50.145) Skip Current_Leg
No
(50.150) EAC>=1?
Yes
(50.220) Create a new P_Series (for Current_Leg) with one Series_Element (COCU =[ ], COEP =[Current_Leg], CSE = 0)
No
(50.190) Legs left in P_LIST?
Yes
No
(50.270) Get U_Series by adding all the Series_Elements from the S_Series of all the Awaiting_Segments
(50.280) Return U_Series
(50.300) End
(50.230) Get S_Series for Current_Segment from P_Series of all Legs
(50.240) Create a new ChildU using Caller_U and add this Child_U to all Series_Elements in the S_Series.
(50.250) Validate S_Series using Qualify_Series_Element Rule
(50.160) Call Upgrade_U for each Ua (in Current Leg) not present in Current COPU, where COPP = COPP of one level up Upgrade_U + Current_Leg, COPU = COPUof one level up Upgrade_U + Current_Ua, Caller_U = Current_Ua, Initiator_Leg = Current_Leg, Initiator_U= Caller_U of one level up Upgrade_U, Benefit = Benefit of highest level Upgrade_U
(50.170) Get the U_Series returned from Upgrade_U for each Ua
(50.180) Get P_Series for Current_Leg by adding all the Series_Elements from all U_Series

Fig. 51

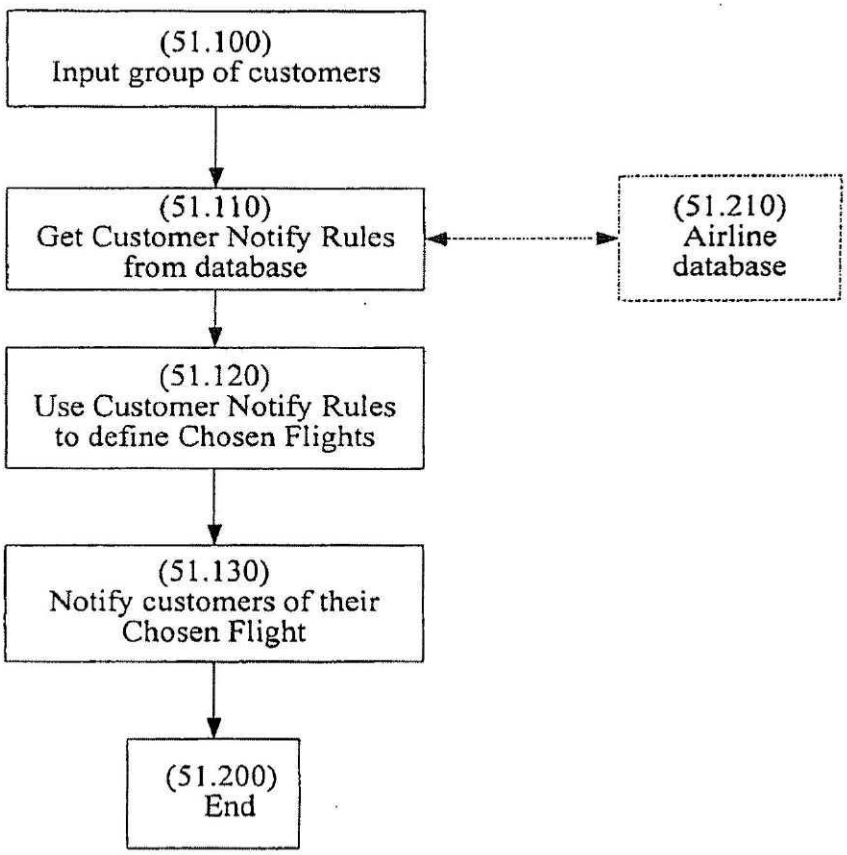

(51.100)
Input group of customers
(51.110)
Get Customer Notify Rules
from database
(51.210)
Airline
database
(51.120)
Use Customer Notify Rules
to define Chosen Flights
(51.130)
Notify customers of their
Chosen Flight
(51.200)
End

Fig. 52

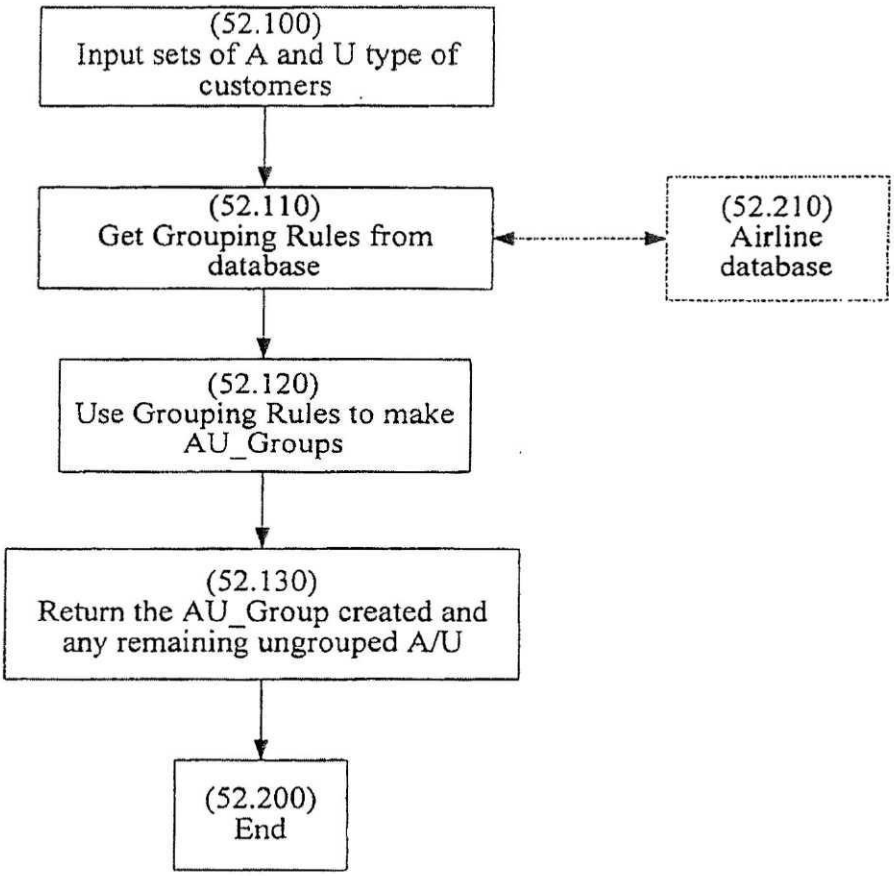

(52.100)
Input sets of A and U type of customers
(52.110)
Get Grouping Rules from database
(52.210)
Airline database
(52.120)
Use Grouping Rules to make AU_Groups
(52.130)
Return the AU_Group created and any remaining ungrouped A/U
(52.200)
End

Fig. 53

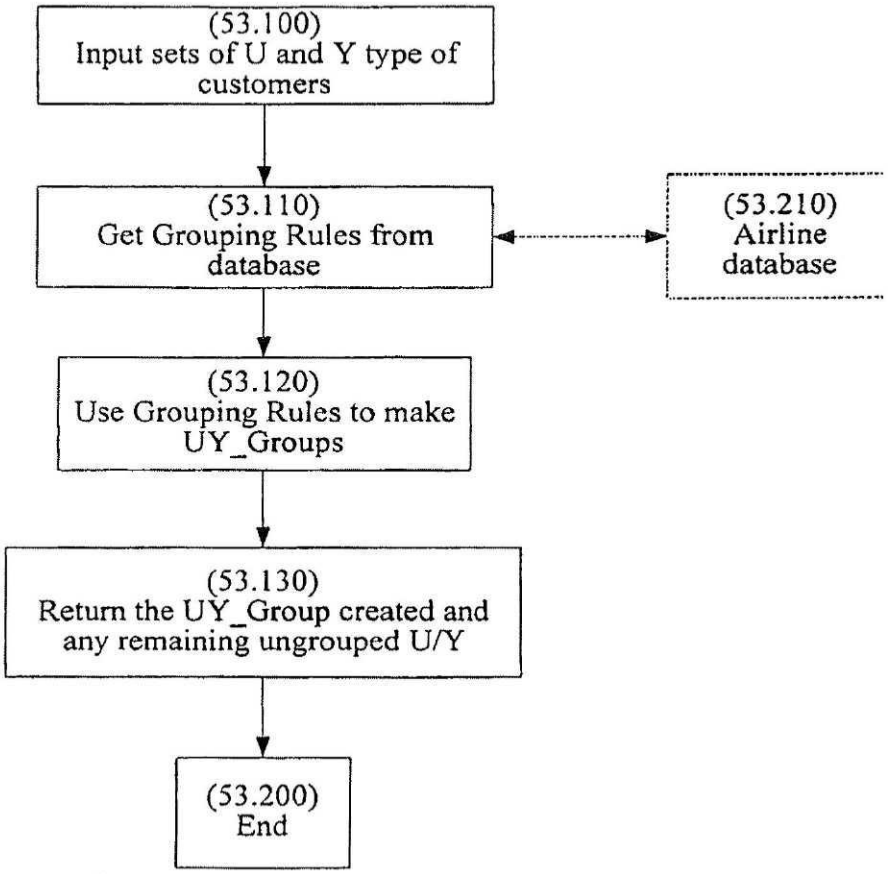

(53.100)
Input sets of U and Y type of customers
(53.110)
Get Grouping Rules from database
(53.210)
Airline database
(53.120)
Use Grouping Rules to make UY_Groups
(53.130)
Return the UY_Group created and any remaining ungrouped U/Y
(53.200)
End

Fig. 54

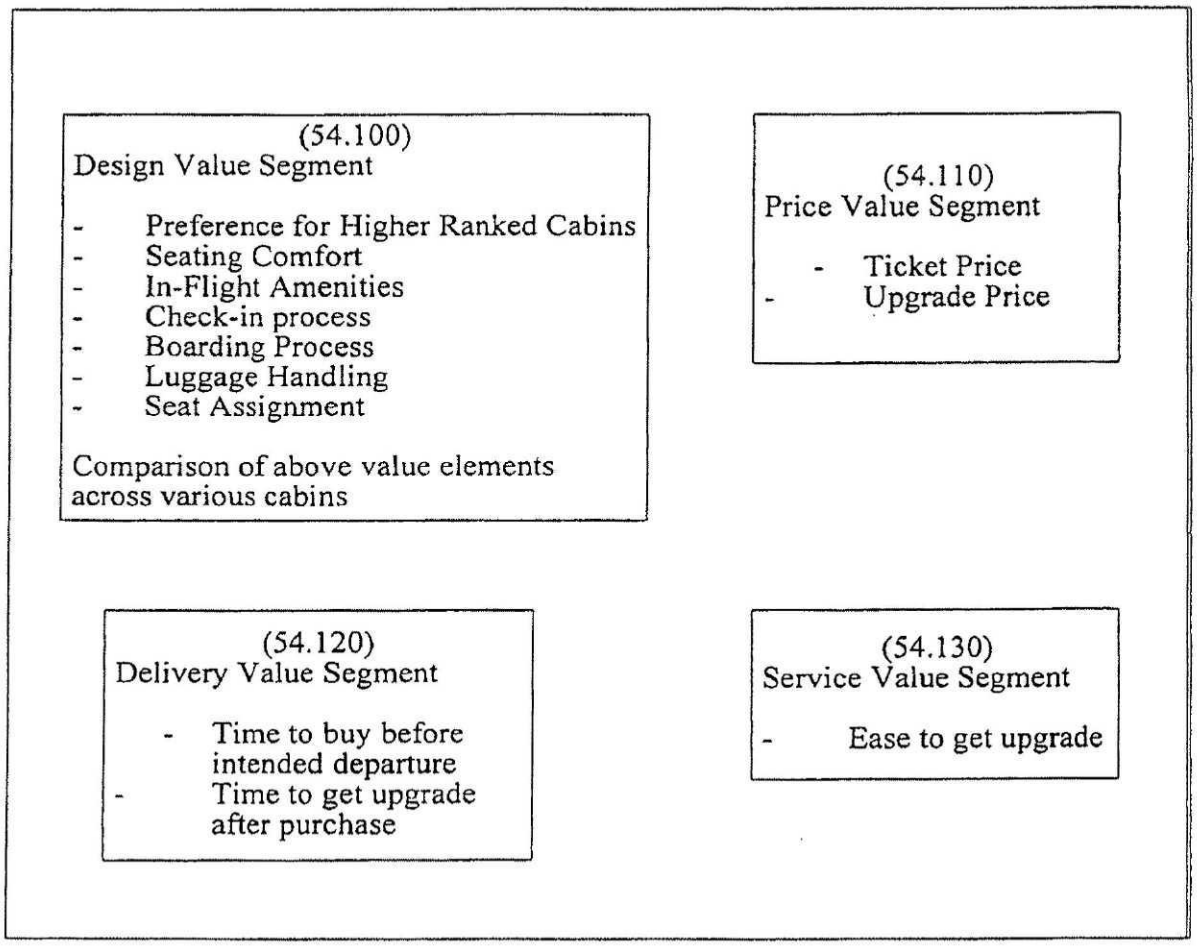

(54.100)
Design Value Segment
- Preference for Higher Ranked Cabins
- Seating Comfort
- In-Flight Amenities
- Check-in process
- Boarding Process
- Luggage Handling
- Seat Assignment
Comparison of above value elements across various cabins
(54.110)
Price Value Segment
- Ticket Price
- Upgrade Price
(54.120)
Delivery Value Segment
- Time to buy before intended departure
- Time to get upgrade after purchase
(54.130)
Service Value Segment
- Ease to get upgrade

Fig. 55

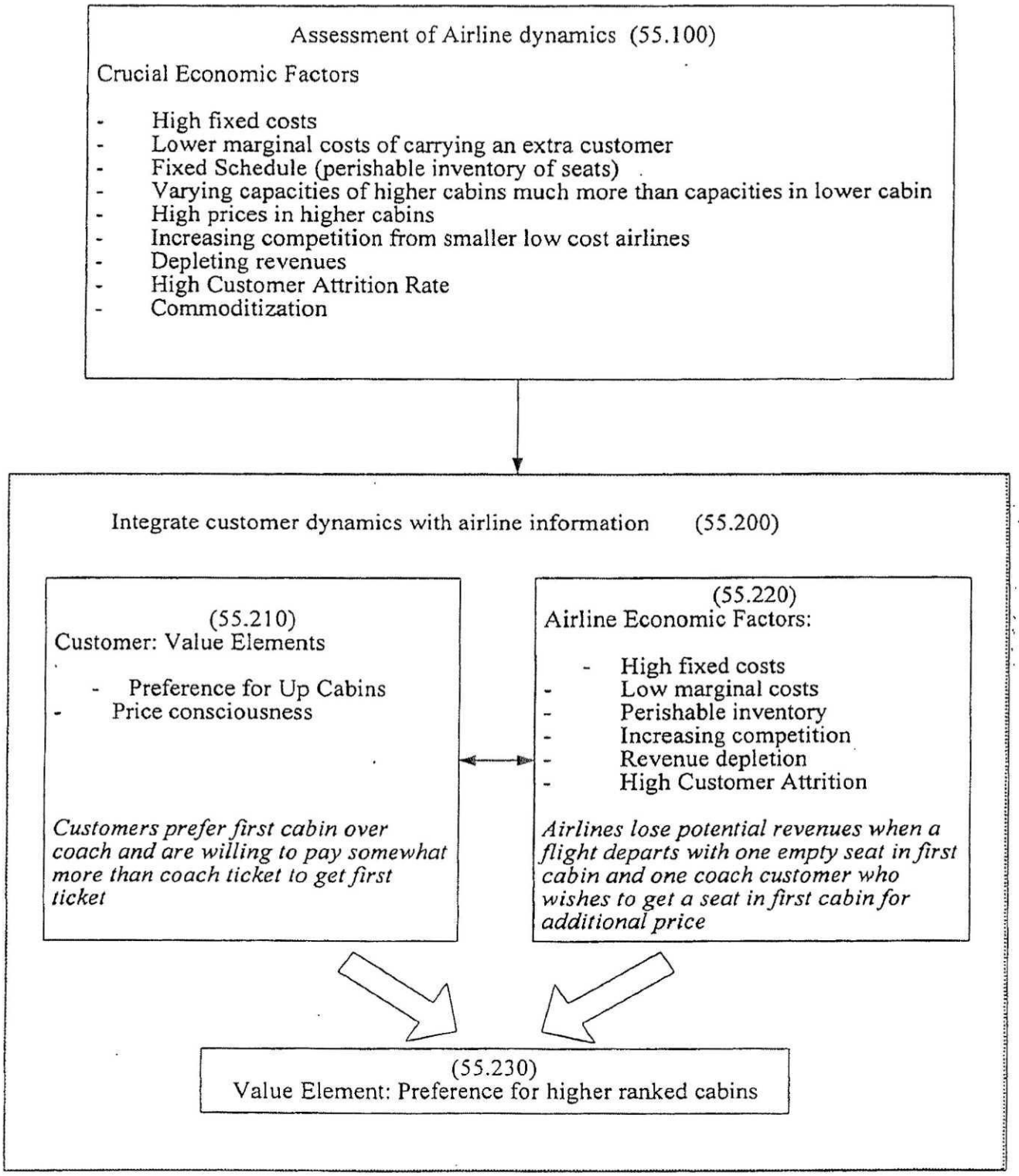

Assessment of Airline dynamics (55.100)
Crucial Economic Factors
- High fixed costs
- Lower marginal costs of carrying an extra customer
- Fixed Schedule (perishable inventory of seats)
- Varying capacities of higher cabins much more than capacities in lower cabin
- High prices in higher cabins
- Increasing competition from smaller low cost airlines
- Depleting revenues
- High Customer Attrition Rate
- Commoditization
Integrate customer dynamics with airline information (55.200)
(55.210)
Customer: Value Elements
- Preference for Up Cabins
- Price consciousness
Customers prefer first cabin over coach and are willing to pay somewhat more than coach ticket to get first ticket
(55.220)
Airline Economic Factors:
- High fixed costs
- Low marginal costs
- Perishable inventory
- Increasing competition
- Revenue depletion
- High Customer Attrition
Airlines lose potential revenues when a flight departs with one empty seat in first cabin and one coach customer who wishes to get a seat in first cabin for additional price
(55.230)
Value Element: Preference for higher ranked cabins

Fig. 56

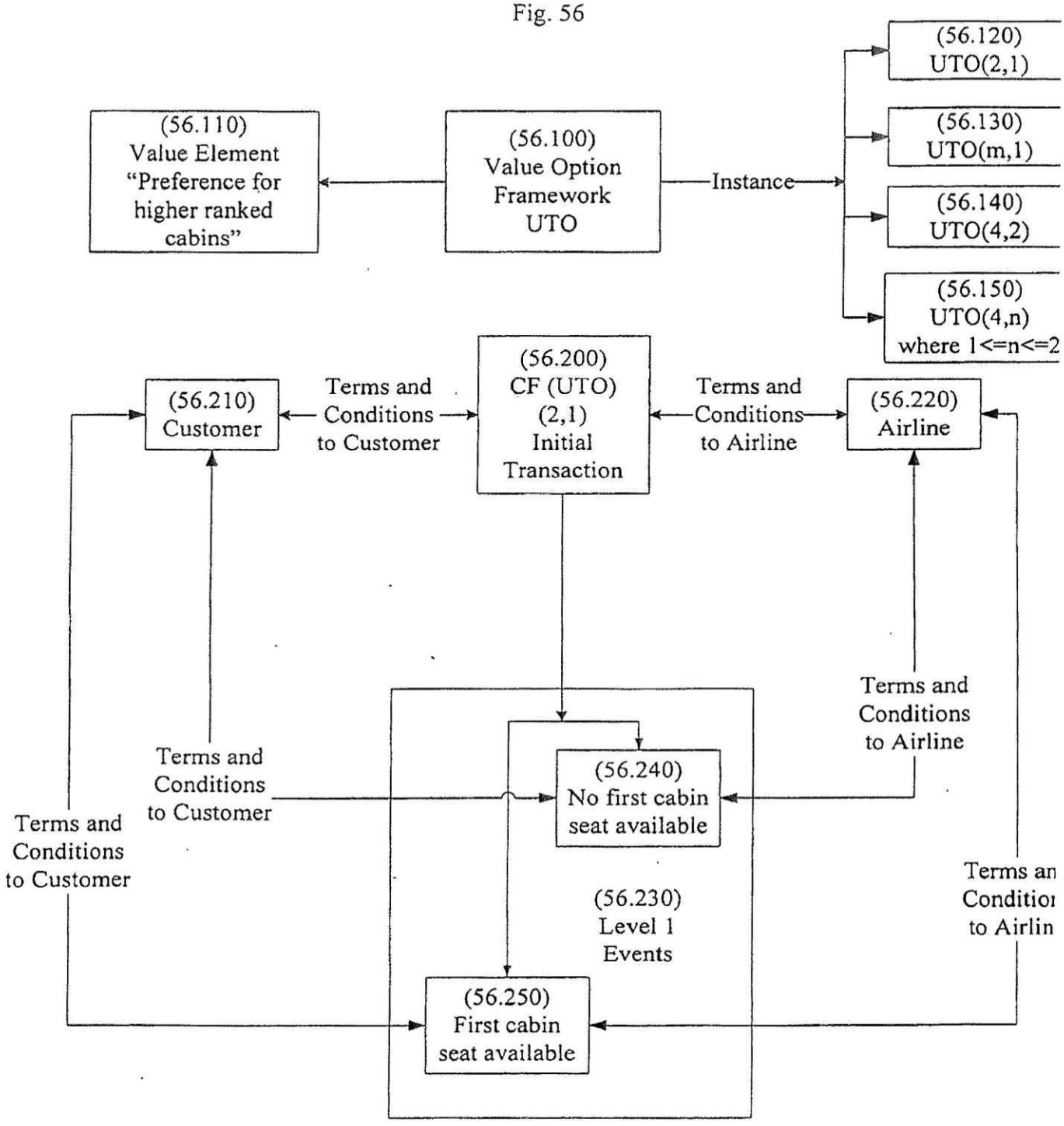

(56.110)
Value Element
"Preference for
higher ranked
cabins"
(56.100)
Value Option
Framework
UTO
Instance
(56.120)
UTO(2,1)
(56.130)
UTO(m,1)
(56.140)
UTO(4,2)
(56.150)
UTO(4,n)
where 1<=n<=2
(56.200)
CF (UTO)
(2,1)
Initial
Transaction
(56.210)
Customer
Terms and
Conditions
to Customer
Terms and
Conditions
to Airline
(56.220)
Airline
Terms and
Conditions
to Customer
Terms and
Conditions
to Customer
(56.240)
No first cabin
seat available
Terms and
Conditions
to Airline
(56.230)
Level 1
Events
(56.250)
First cabin
seat available
Terms an
Conditio
to Airlin

Fig. 57

UTO	Description	Base-Cabin	Up Cabin
CF	Coach to First	Coach	First
CB	Coach to Business	Coach	Business
CBF	Coach to Business or First	Coach	Business, First
BF	Business to First	Business	First

Fig. 58

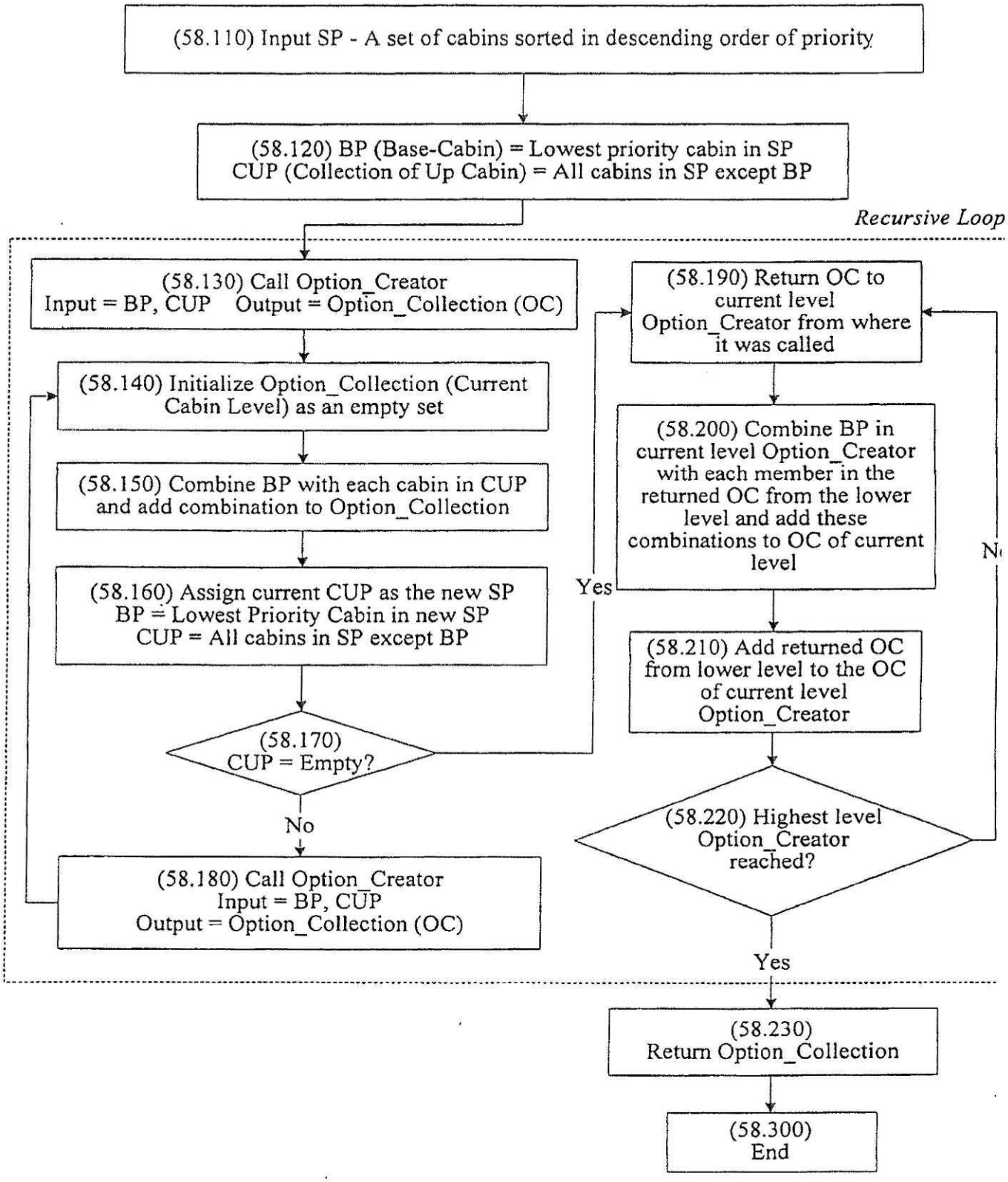

(58.110) Input SP - A set of cabins sorted in descending order of priority
(58.120) BP (Base-Cabin) = Lowest priority cabin in SP
CUP (Collection of Up Cabin) = All cabins in SP except BP
Recursive Loop
(58.130) Call Option_Creator
Input = BP, CUP Output = Option_Collection (OC)
(58.140) Initialize Option_Collection (Current Cabin Level) as an empty set
(58.150) Combine BP with each cabin in CUP and add combination to Option_Collection
(58.160) Assign current CUP as the new SP
BP = Lowest Priority Cabin in new SP
CUP = All cabins in SP except BP
(58.170)
CUP = Empty?
Yes
No
(58.180) Call Option_Creator
Input = BP, CUP
Output = Option_Collection (OC)
(58.190) Return OC to current level Option_Creator from where it was called
(58.200) Combine BP in current level Option_Creator with each member in the returned OC from the lower level and add these combinations to OC of current level
(58.210) Add returned OC from lower level to the OC of current level Option_Creator
(58.220) Highest level Option_Creator reached?
Yes
(58.230)
Return Option_Collection
(58.300)
End

Fig. 59

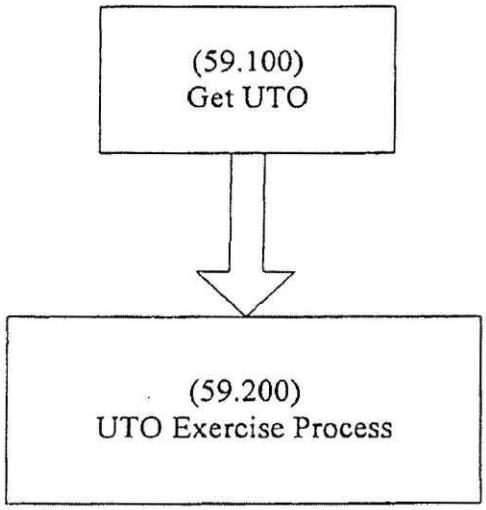


Fig. 60

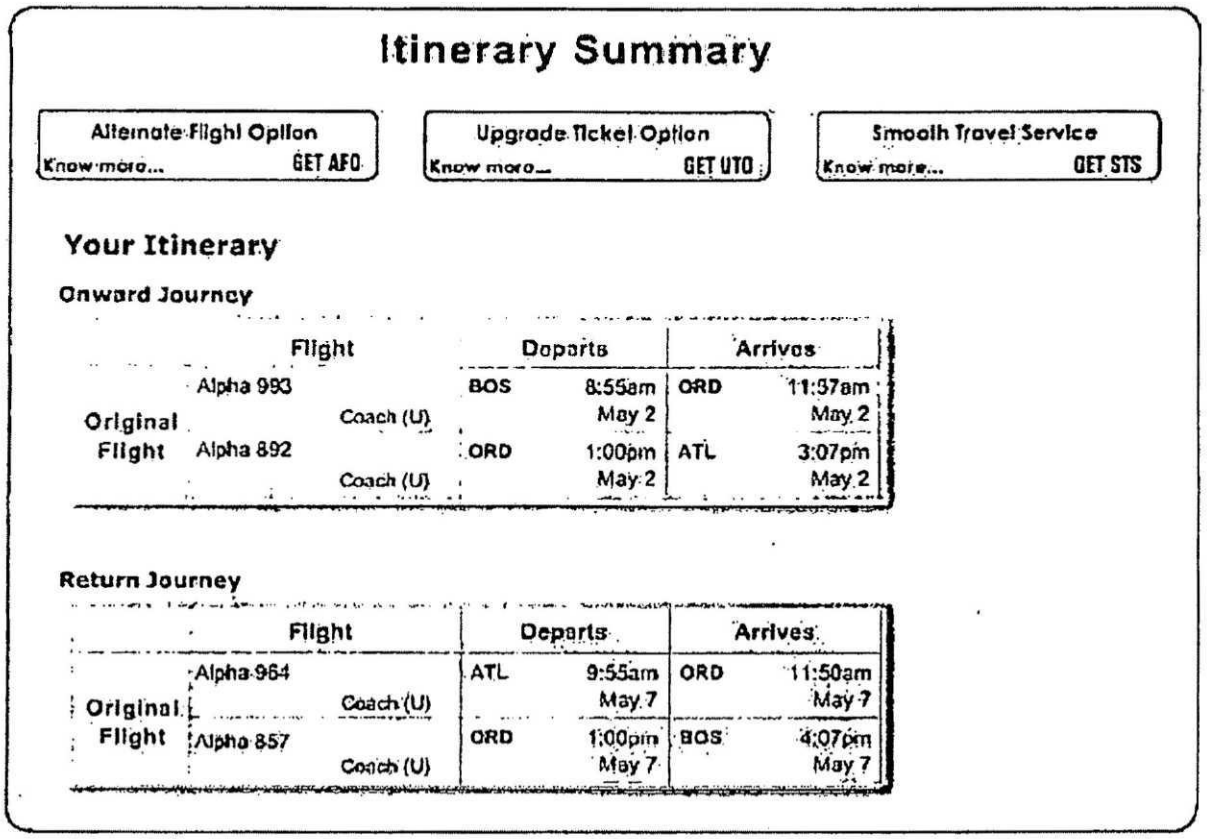

Itinerary Summary
Alternate Flight Option
Know more... GET AFO
Upgrade Ticket Option
Know more... GET UTO
Smooth Travel Service
Know more... GET STS
Your Itinerary
Onward Journey
Flight
Departs
Arrives
Original Flight
Alpha 993
Coach (U)
BOS 8:55am May 2
ORD 11:37am May 2
Alpha 892
Coach (U)
ORD 1:00pm May 2
ATL 3:07pm May 2
Return Journey
Flight
Departs
Arrives
Original Flight
Alpha 964
Coach (U)
ATL 9:55am May 7
ORD 11:50am May 7
Alpha 857
Coach (U)
ORD 1:00pm May 7
BOS 4:07pm May 7

Fig. 61

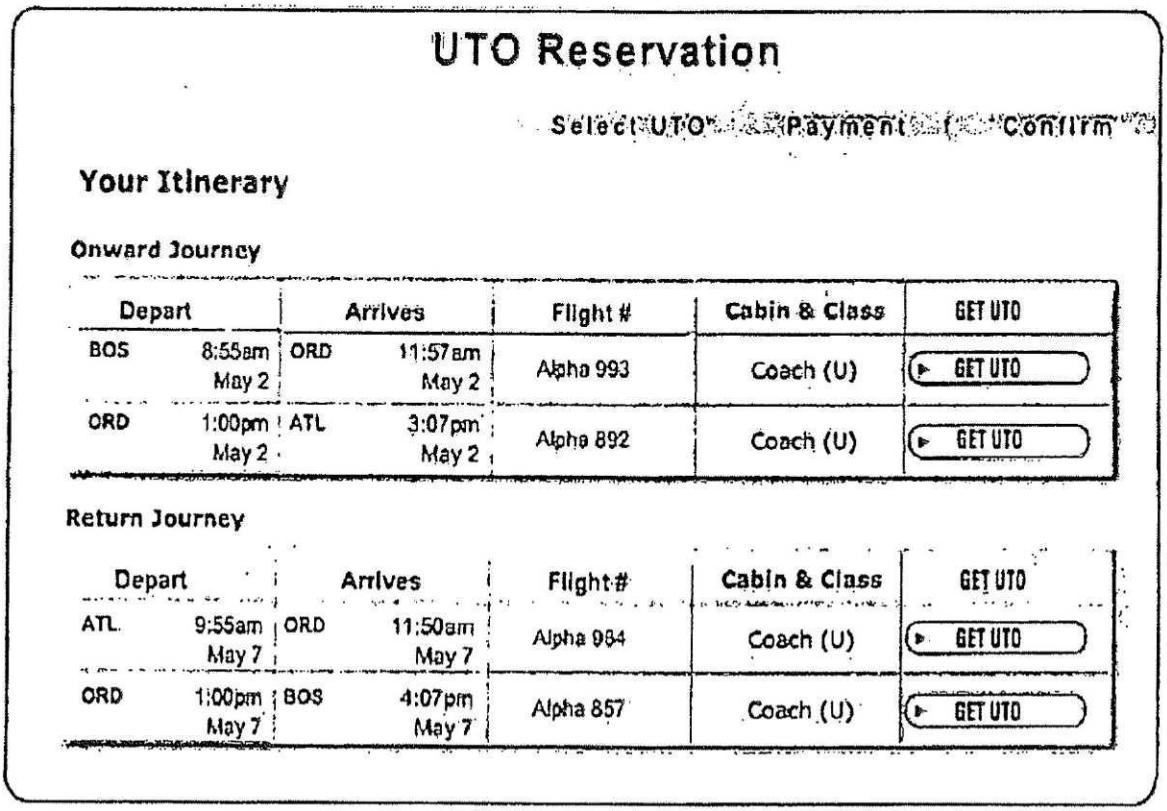

UTO Reservation
Select UTO
Payment
Confirm
Your Itinerary
Onward Journey
Depart
Arrives
Flight #
Cabin & Class
GET UTO
BOS 8:55am May 2
ORD 11:57am May 2
Alpha 993
Coach (U)
GET UTO
ORD 1:00pm May 2
ATL 3:07pm May 2
Alpha 892
Coach (U)
GET UTO
Return Journey
Depart
Arrives
Flight #
Cabin & Class
GET UTO
ATL 9:55am May 7
ORD 11:50am May 7
Alpha 984
Coach (U)
GET UTO
ORD 1:00pm May 7
BOS 4:07pm May 7
Alpha 857
Coach (U)
GET UTO

Fig. 62

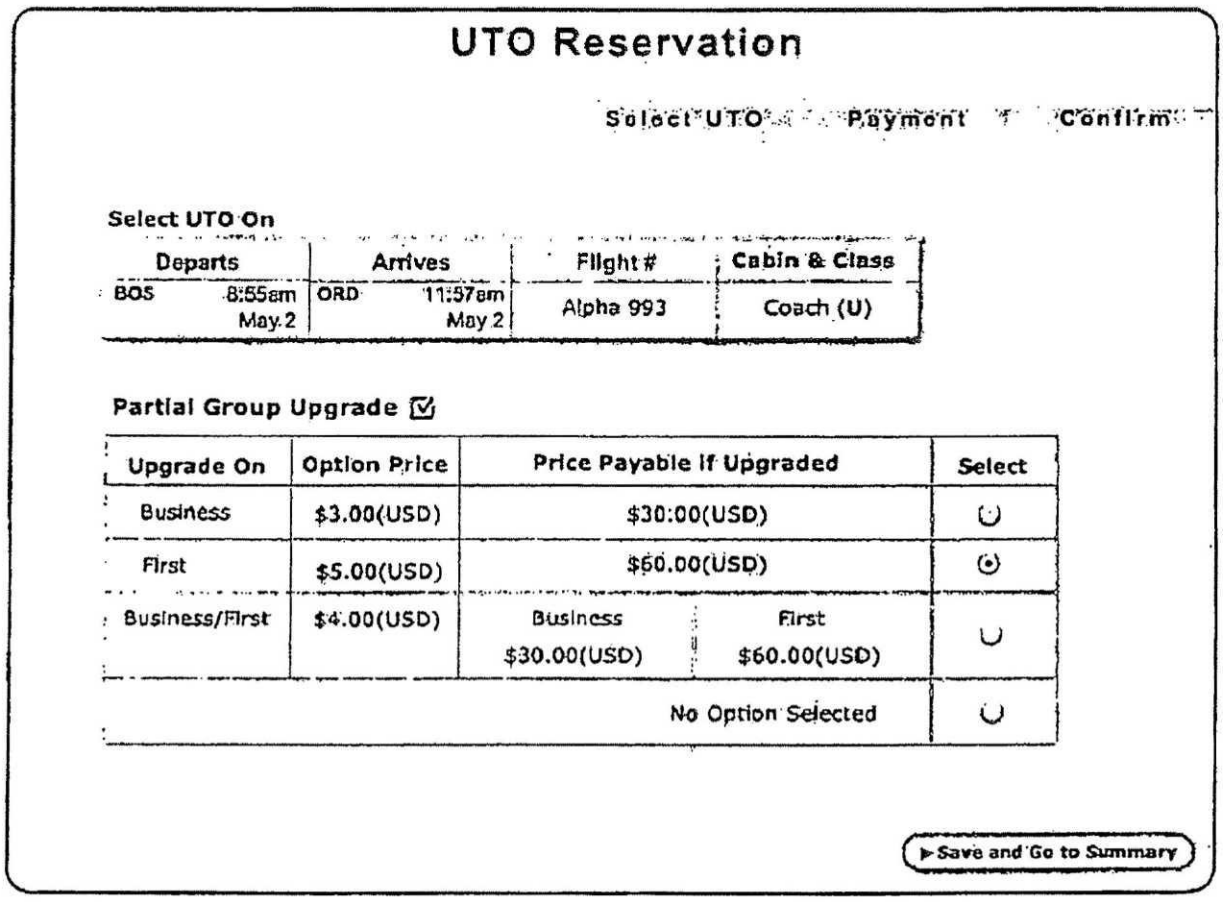

UTO Reservation
Select UTO
Payment
Confirm
Select UTO On
Departs
Arrives
Flight #
Cabin & Class
BOS 8:55am May 2
ORD 11:57am May 2
Alpha 993
Coach (U)
Partial Group Upgrade
Upgrade On
Option Price
Price Payable If Upgraded
Select
Business
$3.00(USD)
$30.00(USD)
First
$5.00(USD)
$60.00(USD)
Business/First
$4.00(USD)
Business $30.00(USD)
First $60.00(USD)
No Option Selected
Save and Go to Summary

Fig. 63

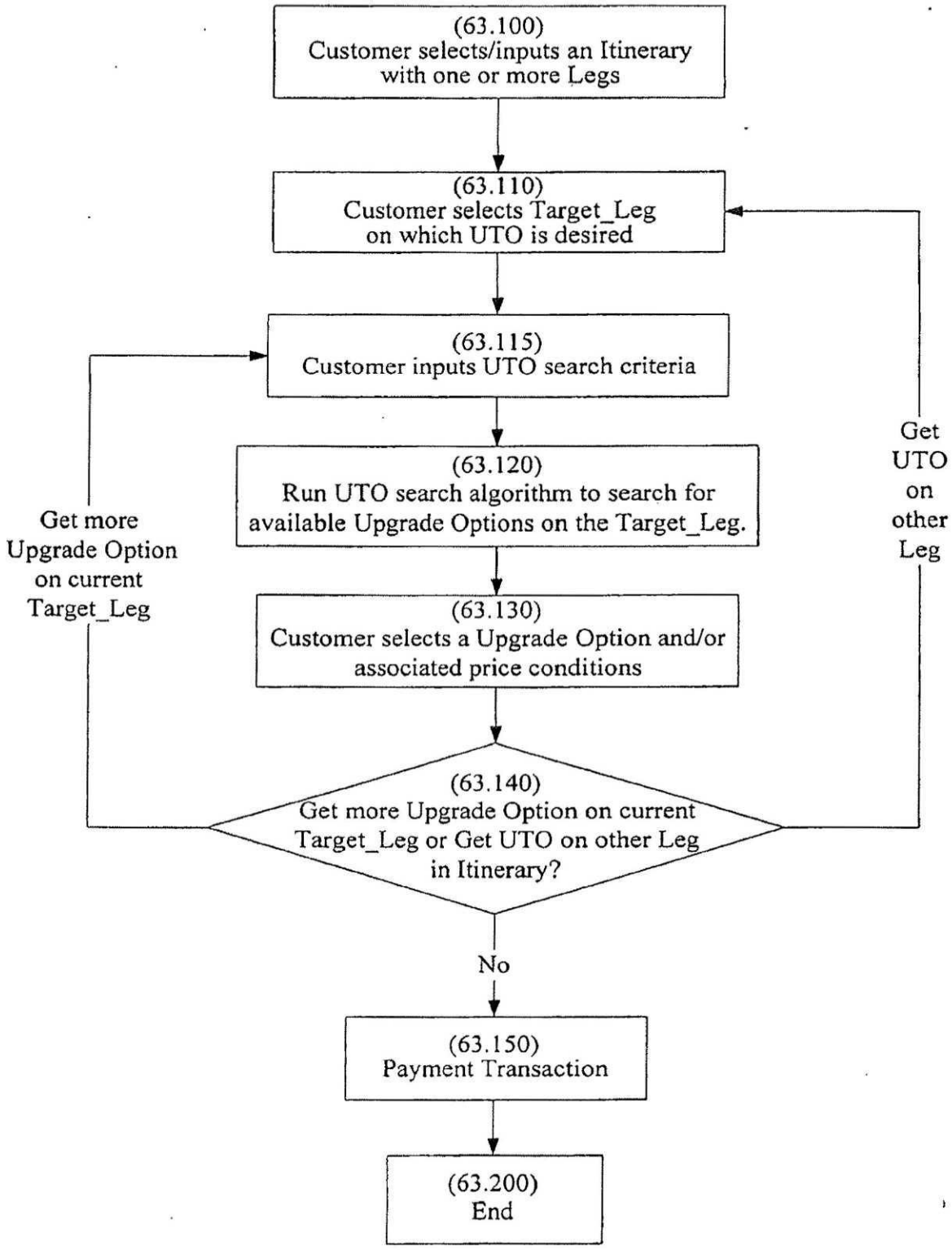

(63.100)
Customer selects/inputs an Itinerary with one or more Legs
(63.110)
Customer selects Target_Leg on which UTO is desired
(63.115)
Customer inputs UTO search criteria
(63.120)
Run UTO search algorithm to search for available Upgrade Options on the Target_Leg.
(63.130)
Customer selects a Upgrade Option and/or associated price conditions
(63.140)
Get more Upgrade Option on current Target_Leg or Get UTO on other Leg in Itinerary?
Get more Upgrade Option on current Target_Leg
Get UTO on other Leg
No
(63.150)
Payment Transaction
(63.200)
End

Fig. 64

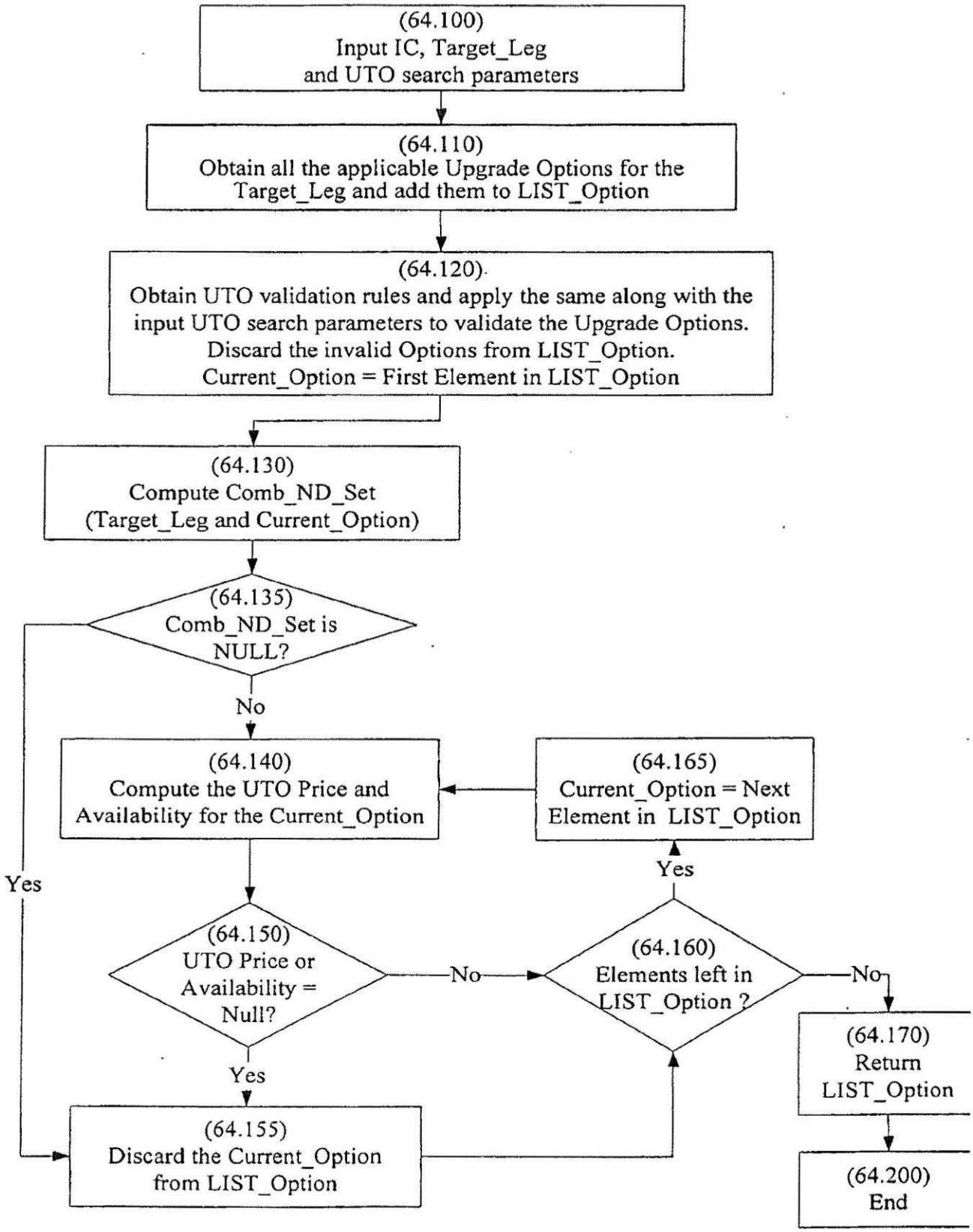

(64.100)
Input IC, Target_Leg
and UTO search parameters
(64.110)
Obtain all the applicable Upgrade Options for the
Target_Leg and add them to LIST_Option
(64.120)
Obtain UTO validation rules and apply the same along with the
input UTO search parameters to validate the Upgrade Options.
Discard the invalid Options from LIST_Option.
Current_Option = First Element in LIST_Option
(64.130)
Compute Comb_ND_Set
(Target_Leg and Current_Option)
(64.135)
Comb_ND_Set is
NULL?
No
Yes
(64.140)
Compute the UTO Price and
Availability for the Current_Option
(64.165)
Current_Option = Next
Element in LIST_Option
Yes
(64.150)
UTO Price or
Availability =
Null?
No
(64.160)
Elements left in
LIST_Option ?
No
Yes
(64.155)
Discard the Current_Option
from LIST_Option
(64.170)
Return
LIST_Option
(64.200)
End

Fig. 65

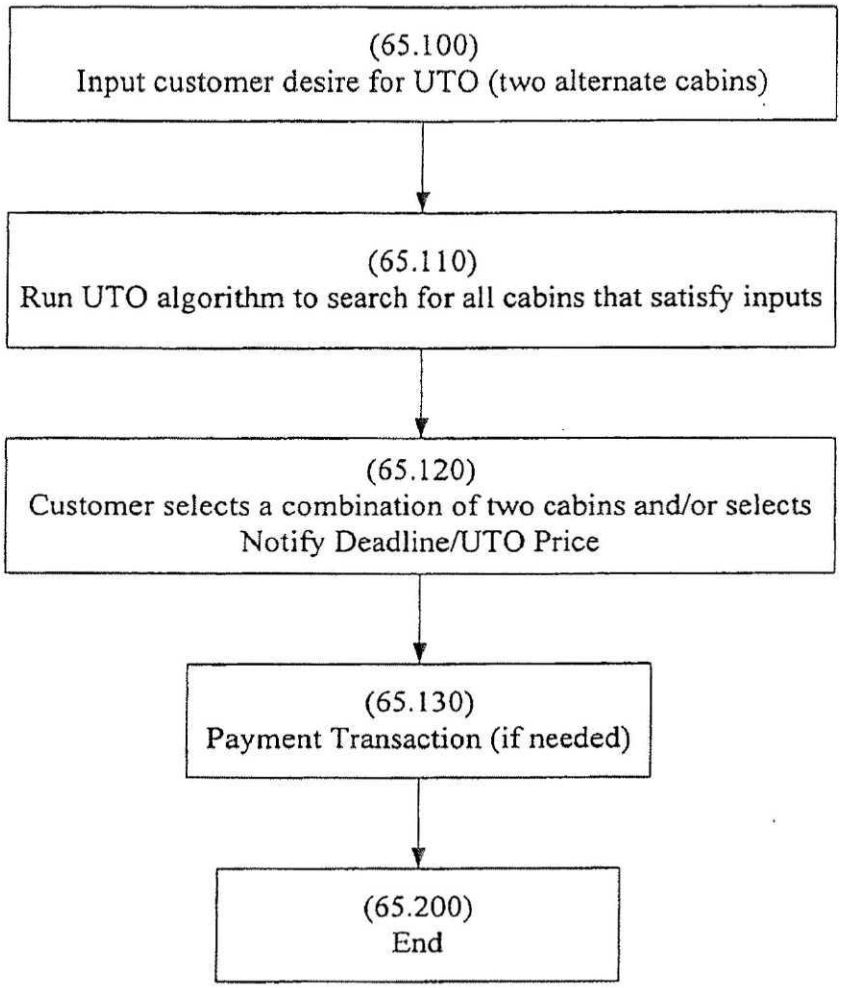

(65.100)
Input customer desire for UTO (two alternate cabins)
(65.110)
Run UTO algorithm to search for all cabins that satisfy inputs
(65.120)
Customer selects a combination of two cabins and/or selects
Notify Deadline/UTO Price
(65.130)
Payment Transaction (if needed)
(65.200)
End

Fig. 66

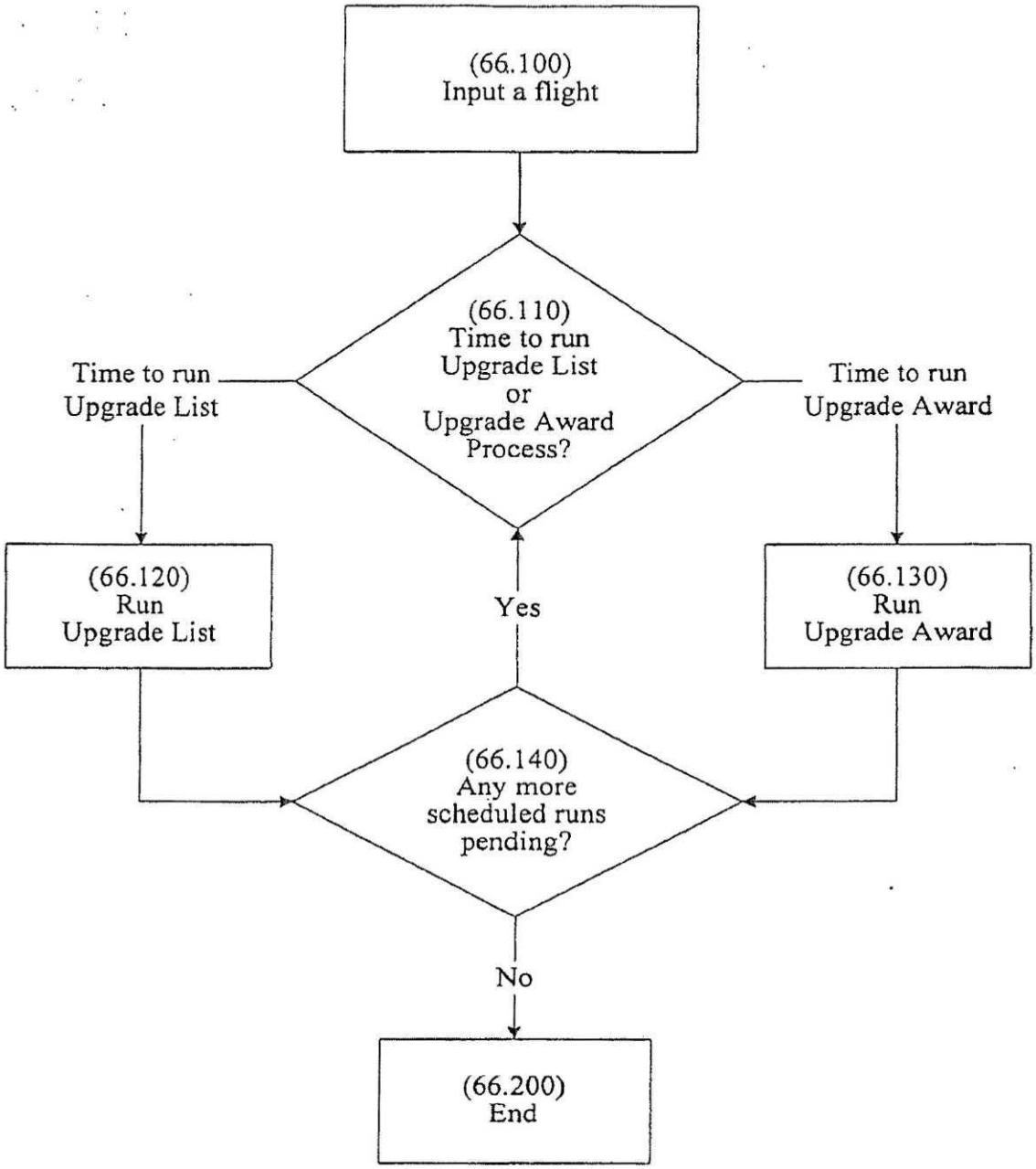

(66.100)
Input a flight
(66.110)
Time to run
Upgrade List
or
Upgrade Award
Process?
Time to run
Upgrade List
Time to run
Upgrade Award
(66.120)
Run
Upgrade List
Yes
(66.130)
Run
Upgrade Award
(66.140)
Any more
scheduled runs
pending?
No
(66.200)
End

Fig. 67

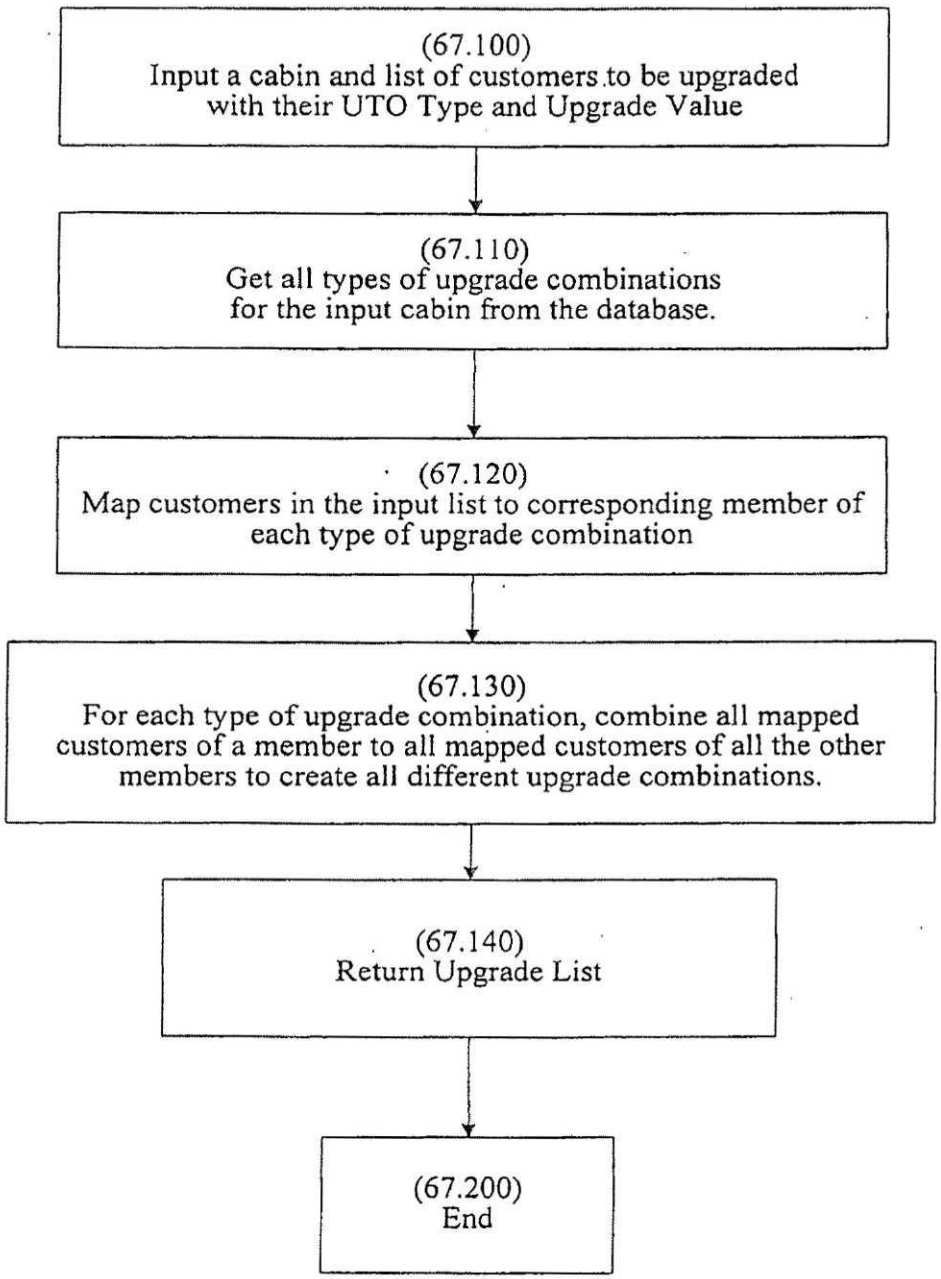

(67.100)
Input a cabin and list of customers to be upgraded with their UTO Type and Upgrade Value
(67.110)
Get all types of upgrade combinations for the input cabin from the database.
(67.120)
Map customers in the input list to corresponding member of each type of upgrade combination
(67.130)
For each type of upgrade combination, combine all mapped customers of a member to all mapped customers of all the other members to create all different upgrade combinations.
(67.140)
Return Upgrade List
(67.200)
End

Fig. 68

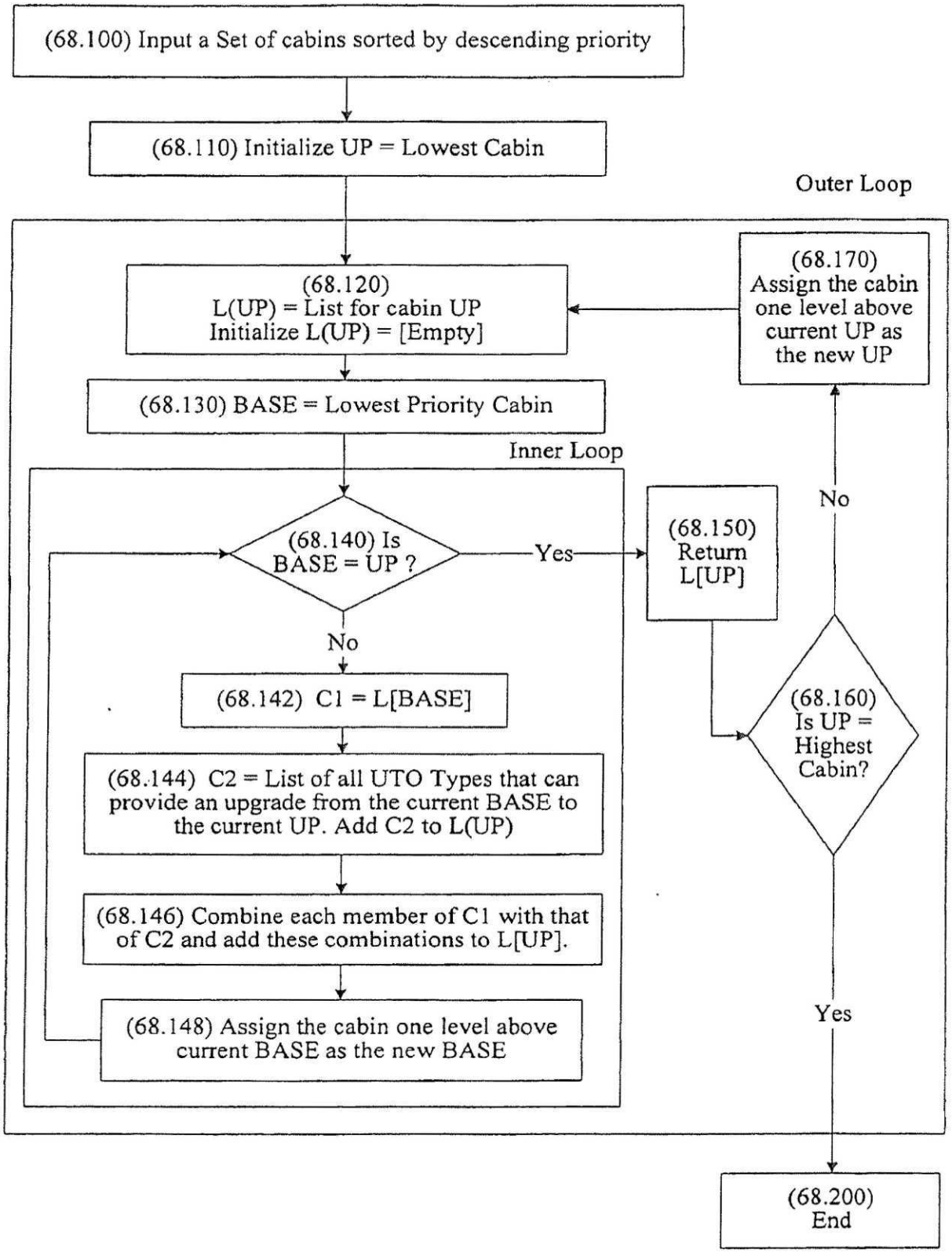

(68.100) Input a Set of cabins sorted by descending priority
(68.110) Initialize UP = Lowest Cabin
Outer Loop
(68.120) L(UP) = List for cabin UP Initialize L(UP) = [Empty]
(68.170) Assign the cabin one level above current UP as the new UP
(68.130) BASE = Lowest Priority Cabin
Inner Loop
(68.140) Is BASE = UP ?
Yes
(68.150) Return L[UP]
No
No
(68.142) C1 = L[BASE]
(68.160) Is UP = Highest Cabin?
(68.144) C2 = List of all UTO Types that can provide an upgrade from the current BASE to the current UP. Add C2 to L(UP)
(68.146) Combine each member of C1 with that of C2 and add these combinations to L[UP].
(68.148) Assign the cabin one level above current BASE as the new BASE
Yes
(68.200) End

Fig. 69

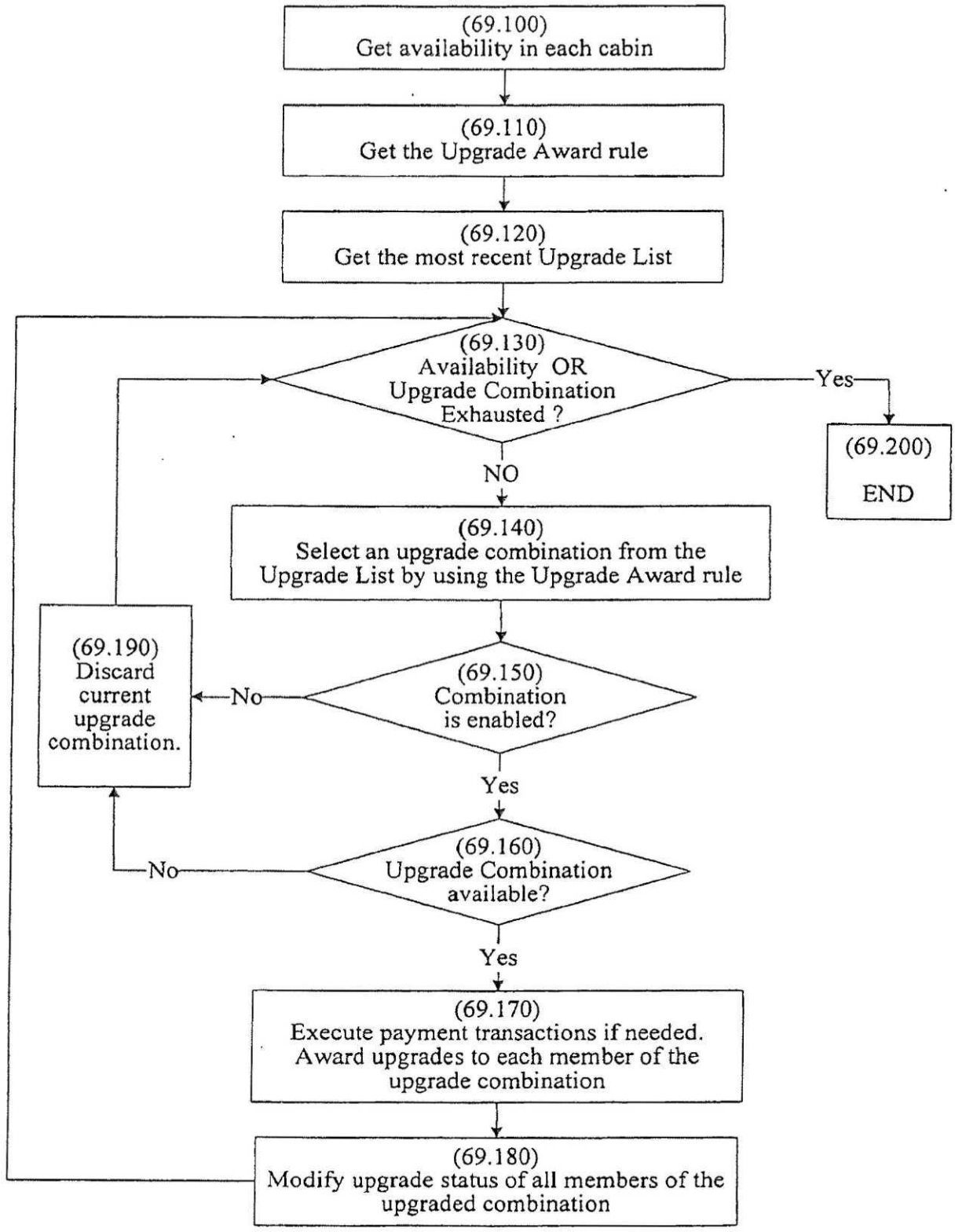
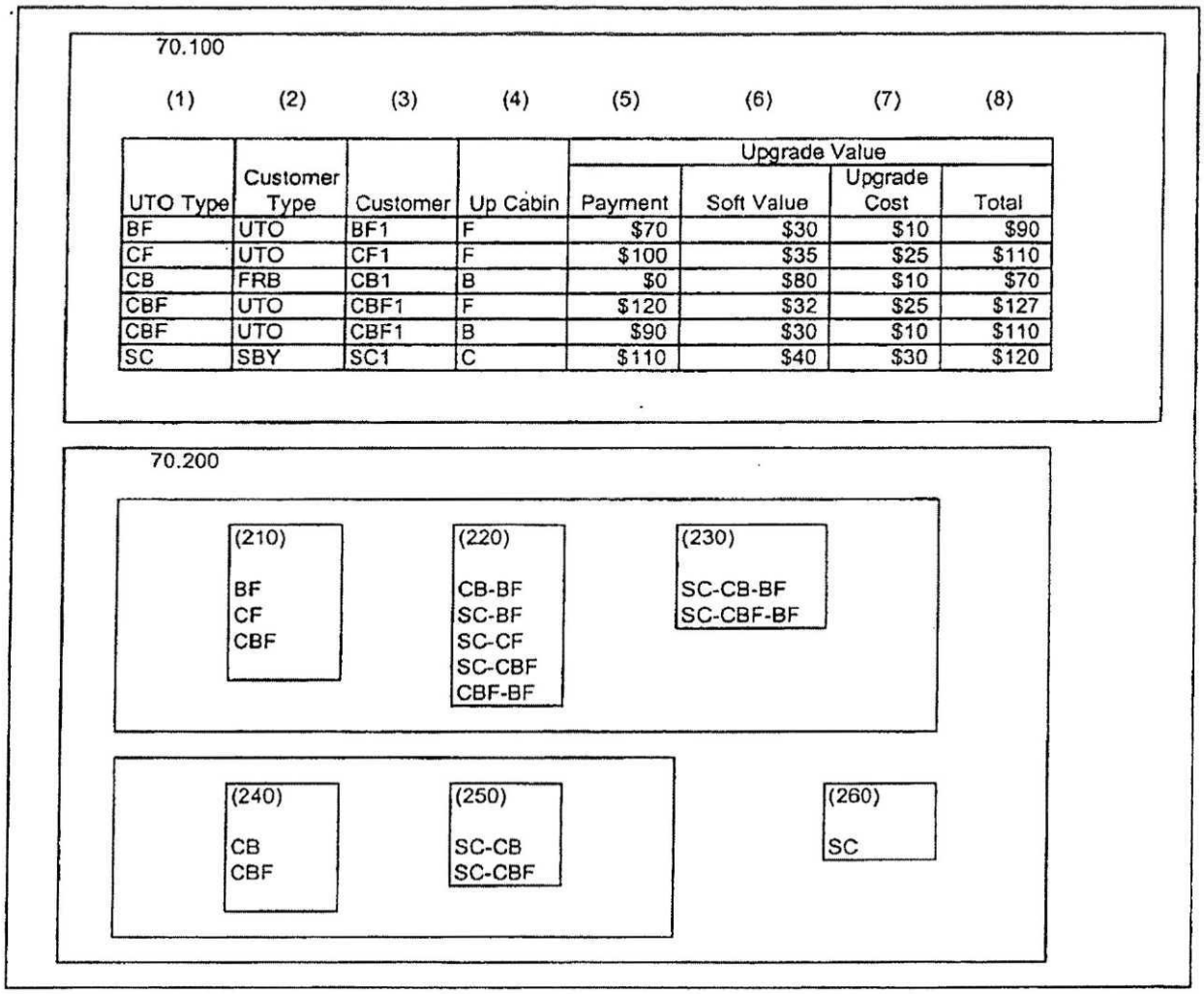

(69.100)
Get availability in each cabin
(69.110)
Get the Upgrade Award rule
(69.120)
Get the most recent Upgrade List
(69.130)
Availability OR
Upgrade Combination
Exhausted ?
Yes
(69.200)
END
NO
(69.140)
Select an upgrade combination from the
Upgrade List by using the Upgrade Award rule
(69.150)
Combination
is enabled?
No
(69.190)
Discard
current
upgrade
combination.
Yes
(69.160)
Upgrade Combination
available?
No
Yes
(69.170)
Execute payment transactions if needed.
Award upgrades to each member of the
upgrade combination
(69.180)
Modify upgrade status of all members of the
upgraded combination

Fig. 70



UTO Type	Customer Type	Customer	Up Cabin	Upgrade Value			
				Payment	Soft Value	Upgrade Cost	Total
BF	UTO	BF1	F	$70	$30	$10	$90
CF	UTO	CF1	F	$100	$35	$25	$110
CB	FRB	CB1	B	$0	$80	$10	$70
CBF	UTO	CBF1	F	$120	$32	$25	$127
CBF	UTO	CBF1	B	$90	$30	$10	$110
SC	SBY	SC1	C	$110	$40	$30	$120

Fig. 71

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
S. No.	First	Business	Coach	Value(F)	Value(B)	Value(C)	Total Value
1	BF1	CBF1	SC1	$90	$110	$120	$320
2	BF1	CB1	SC1	$90	$70	$120	$280
3	CBF1		SC1	$127		$120	$247
4	CF1		SC1	$110		$120	$230
5		*CBF1*	*SC1*		$110	$120	$230
6	BF1		SC1	$90		$120	$210
7	BF1	CBF1		$90	$110		$200
8		*CB1*	*SC1*		$70	$120	$190
9	BF1	CB1		$90	$70		$160
10	CBF1			$127			$127
11			SC1			$120	$120
12	CF1			$110			$110
13		*CBF1*			$110		$110
14	BF1			$90			$90
15		*CB1*			$70		$70

Fig. 72

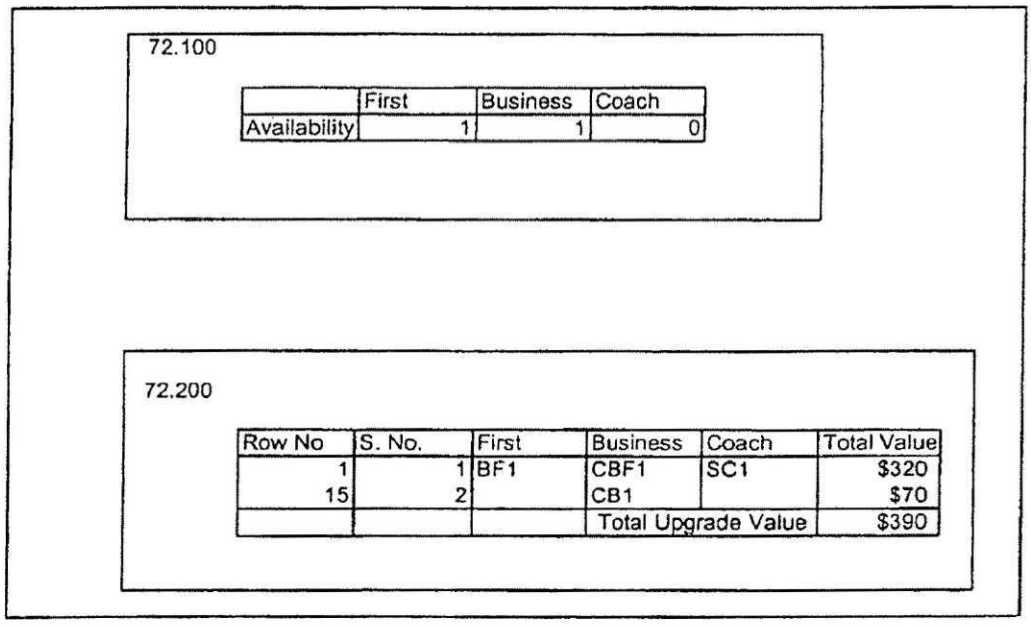


Fig. 73

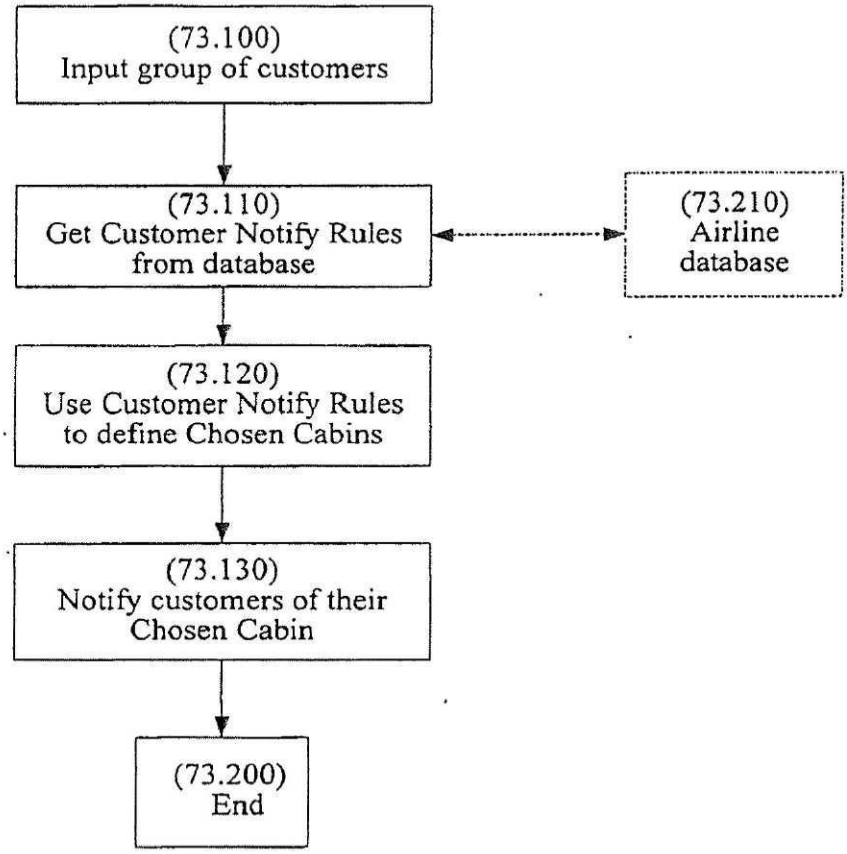

(73.100)
Input group of customers
(73.110)
Get Customer Notify Rules
from database
(73.210)
Airline
database
(73.120)
Use Customer Notify Rules
to define Chosen Cabins
(73.130)
Notify customers of their
Chosen Cabin
(73.200)
End

Fig. 74

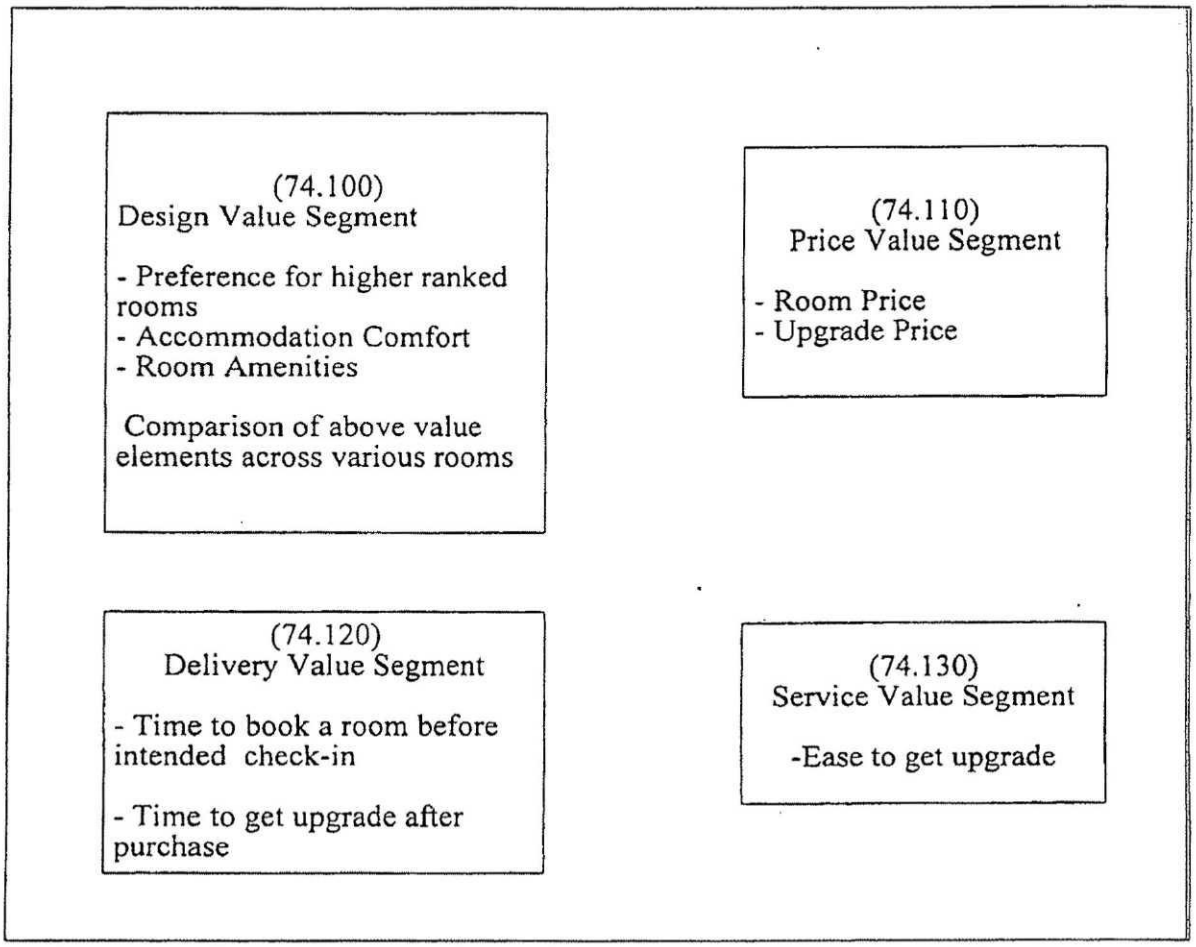

(74.100)
Design Value Segment
- Preference for higher ranked rooms
- Accommodation Comfort
- Room Amenities
Comparison of above value elements across various rooms
(74.110)
Price Value Segment
- Room Price
- Upgrade Price
(74.120)
Delivery Value Segment
- Time to book a room before intended check-in
- Time to get upgrade after purchase
(74.130)
Service Value Segment
-Ease to get upgrade

Fig. 75

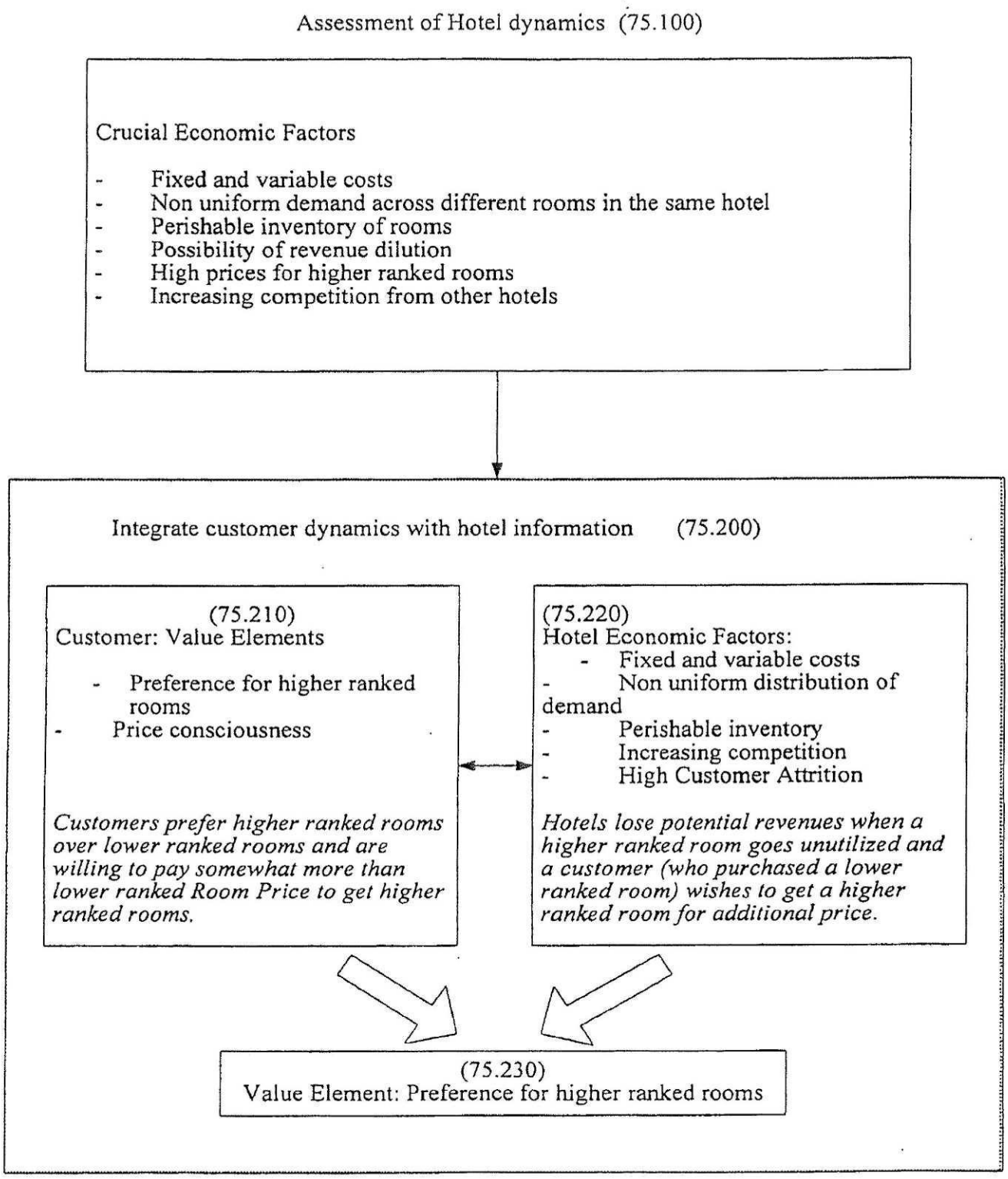

Assessment of Hotel dynamics (75.100)
Crucial Economic Factors
- Fixed and variable costs
- Non uniform demand across different rooms in the same hotel
- Perishable inventory of rooms
- Possibility of revenue dilution
- High prices for higher ranked rooms
- Increasing competition from other hotels
Integrate customer dynamics with hotel information (75.200)
(75.210)
Customer: Value Elements
- Preference for higher ranked rooms
- Price consciousness
Customers prefer higher ranked rooms over lower ranked rooms and are willing to pay somewhat more than lower ranked Room Price to get higher ranked rooms.
(75.220)
Hotel Economic Factors:
- Fixed and variable costs
- Non uniform distribution of demand
- Perishable inventory
- Increasing competition
- High Customer Attrition
Hotels lose potential revenues when a higher ranked room goes unutilized and a customer (who purchased a lower ranked room) wishes to get a higher ranked room for additional price.
(75.230)
Value Element: Preference for higher ranked rooms

Fig. 76

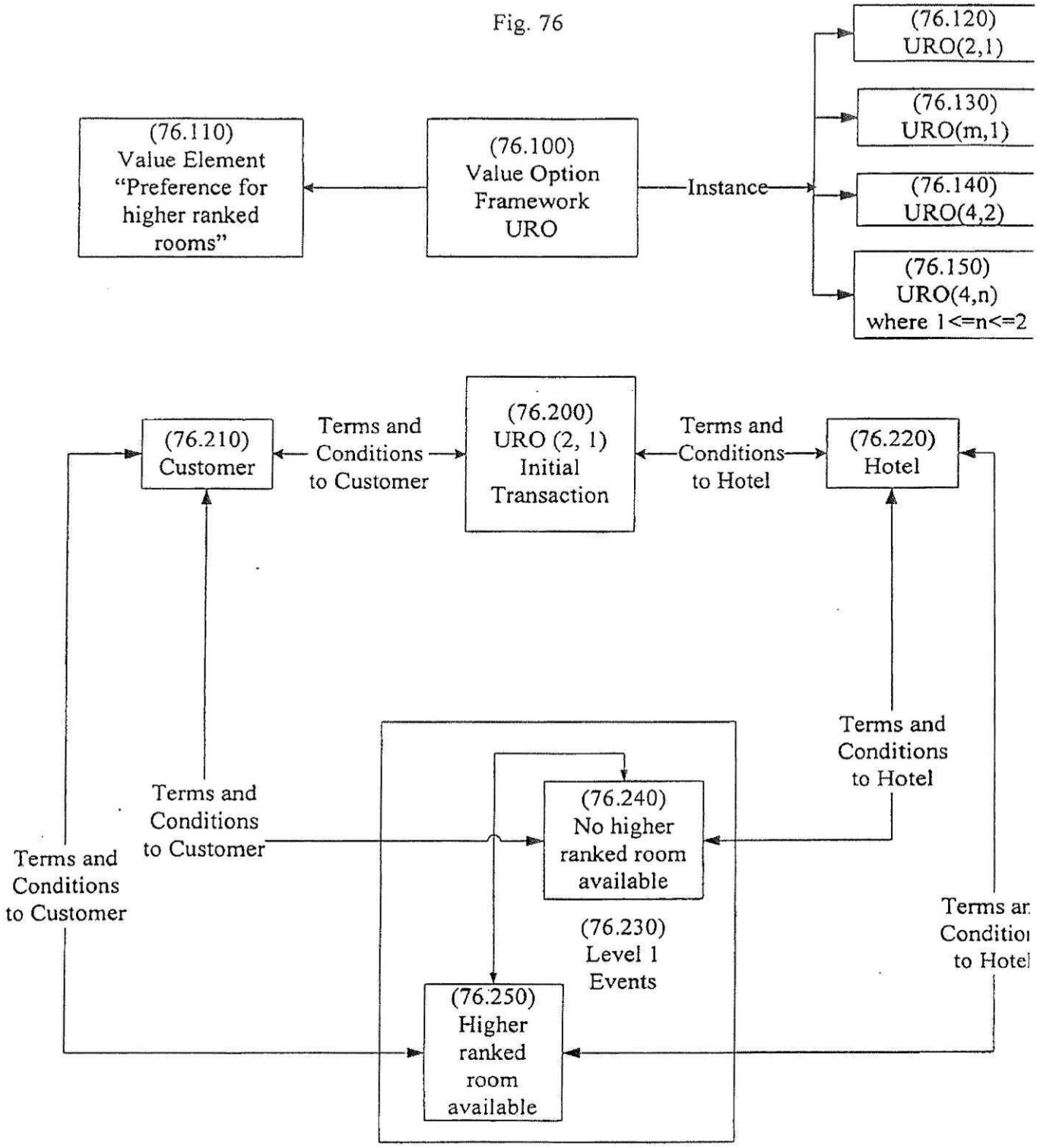

(76.110)
Value Element
"Preference for higher ranked rooms"
(76.100)
Value Option Framework URO
Instance
(76.120)
URO(2,1)
(76.130)
URO(m,1)
(76.140)
URO(4,2)
(76.150)
URO(4,n)
where 1<=n<=2
(76.210)
Customer
Terms and Conditions to Customer
(76.200)
URO (2, 1)
Initial Transaction
Terms and Conditions to Hotel
(76.220)
Hotel
Terms and Conditions to Customer
Terms and Conditions to Customer
Terms and Conditions to Hotel
(76.240)
No higher ranked room available
(76.230)
Level 1 Events
(76.250)
Higher ranked room available
Terms ar
Conditio
to Hotel

Fig. 77

URO	Description	base-room	up Room
GV	Guest to VIP	Guest	VIP
GB	Guest to Business	Guest	Business
GBV	Guest to Business or VIP	Guest	Business, VIP
BV	Business to VIP	Business	VIP

Fig. 78

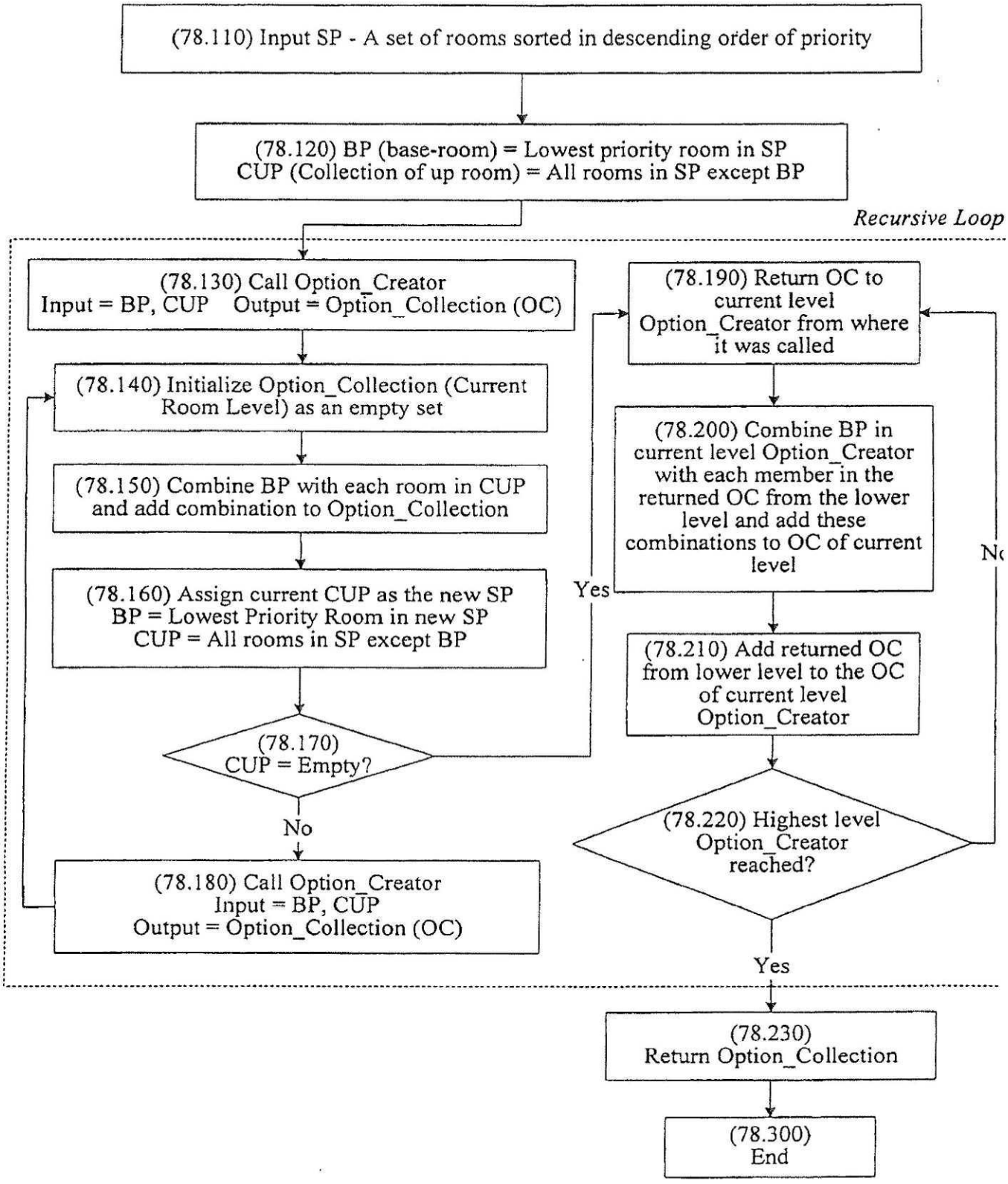

(78.110) Input SP - A set of rooms sorted in descending order of priority
(78.120) BP (base-room) = Lowest priority room in SP
CUP (Collection of up room) = All rooms in SP except BP
Recursive Loop
(78.130) Call Option_Creator
Input = BP, CUP Output = Option_Collection (OC)
(78.140) Initialize Option_Collection (Current Room Level) as an empty set
(78.150) Combine BP with each room in CUP and add combination to Option_Collection
(78.160) Assign current CUP as the new SP
BP = Lowest Priority Room in new SP
CUP = All rooms in SP except BP
(78.170) CUP = Empty?
Yes
No
(78.180) Call Option_Creator
Input = BP, CUP
Output = Option_Collection (OC)
(78.190) Return OC to current level Option_Creator from where it was called
(78.200) Combine BP in current level Option_Creator with each member in the returned OC from the lower level and add these combinations to OC of current level
(78.210) Add returned OC from lower level to the OC of current level Option_Creator
(78.220) Highest level Option_Creator reached?
No
Yes
(78.230) Return Option_Collection
(78.300) End

Fig. 79

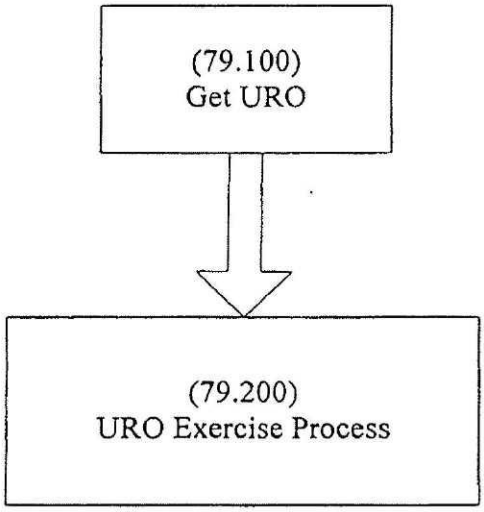


Fig. 80

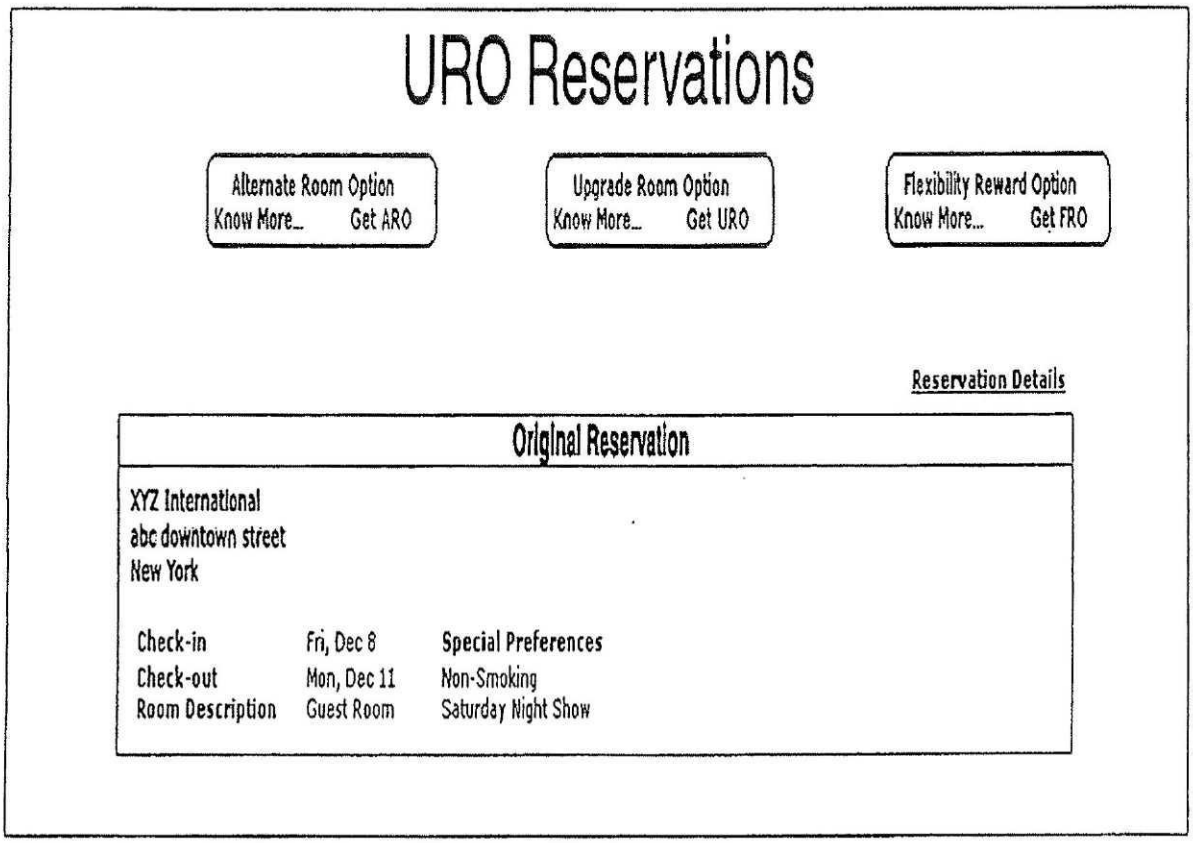

URO Reservations
Alternate Room Option
Know More... Get ARO
Upgrade Room Option
Know More... Get URO
Flexibility Reward Option
Know More... Get FRO
Reservation Details
Original Reservation
XYZ International
abc downtown street
New York
Check-in Fri, Dec 8
Check-out Mon, Dec 11
Room Description Guest Room
Special Preferences
Non-Smoking
Saturday Night Show

Fig. 81

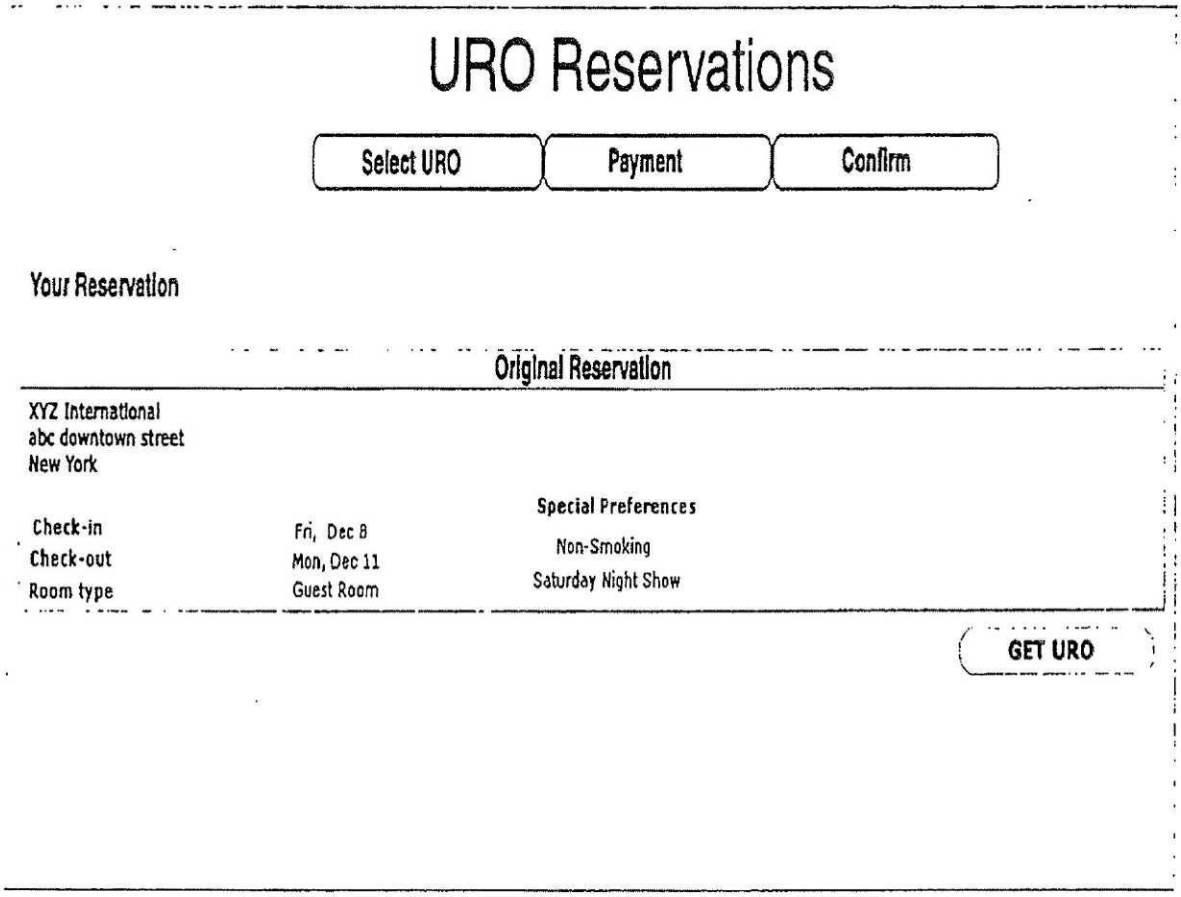

URO Reservations
Select URO
Payment
Confirm
Your Reservation
Original Reservation
XYZ International
abc downtown street
New York
Special Preferences
Check-in
Fri, Dec 8
Non-Smoking
Check-out
Mon, Dec 11
Saturday Night Show
Room type
Guest Room
GET URO

Fig. 82

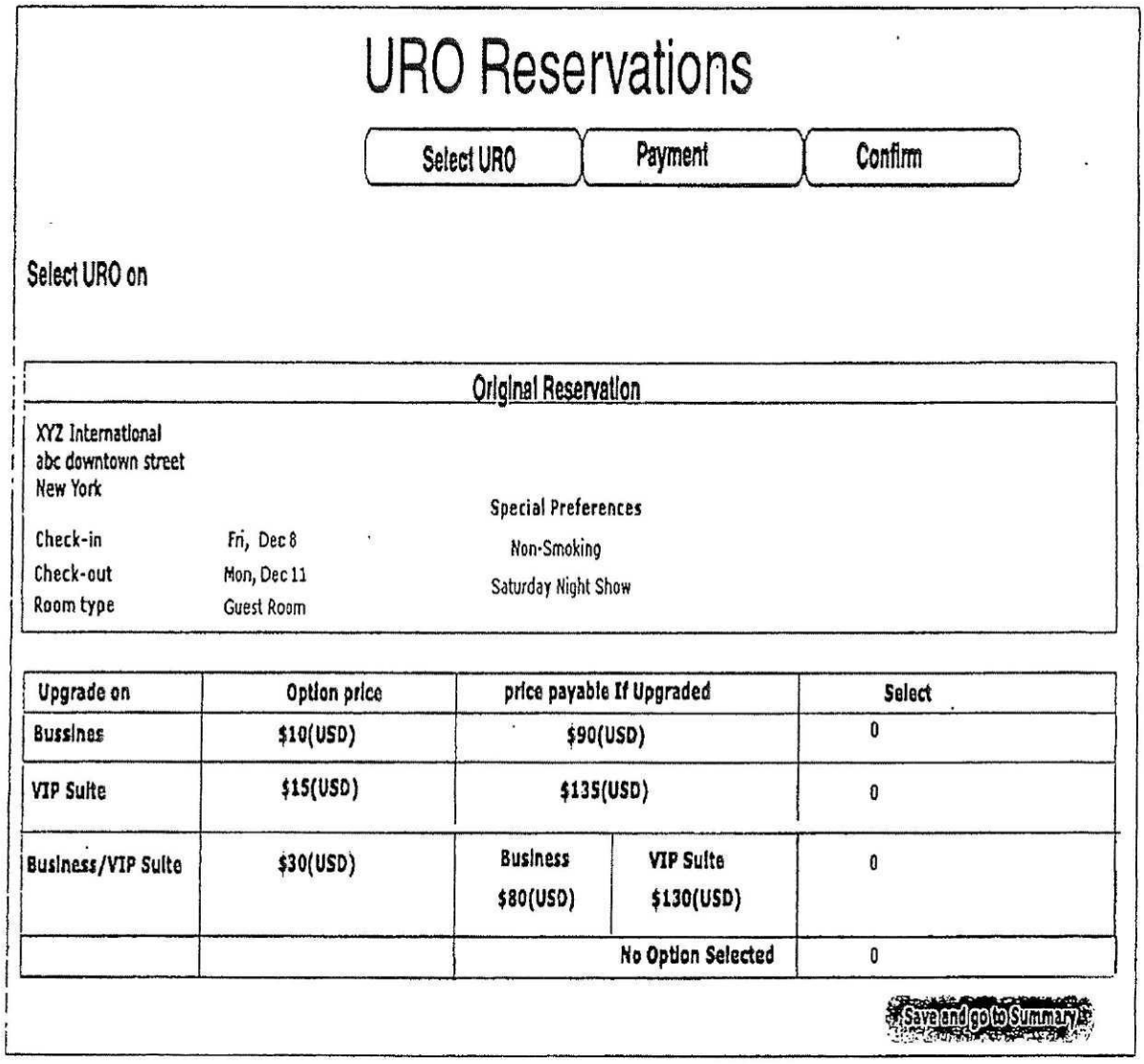

URO Reservations
Select URO
Payment
Confirm
Select URO on
Original Reservation
XYZ International
abc downtown street
New York
Special Preferences
Check-in
Fri, Dec 8
Non-Smoking
Check-out
Mon, Dec 11
Saturday Night Show
Room type
Guest Room
Upgrade on
Option price
price payable If Upgraded
Select
Bussines
$10(USD)
$90(USD)
0
VIP Suite
$15(USD)
$135(USD)
0
Business/VIP Suite
$30(USD)
Business $80(USD)
VIP Suite $130(USD)
0
No Option Selected
0
Save and go to Summary

Fig. 83

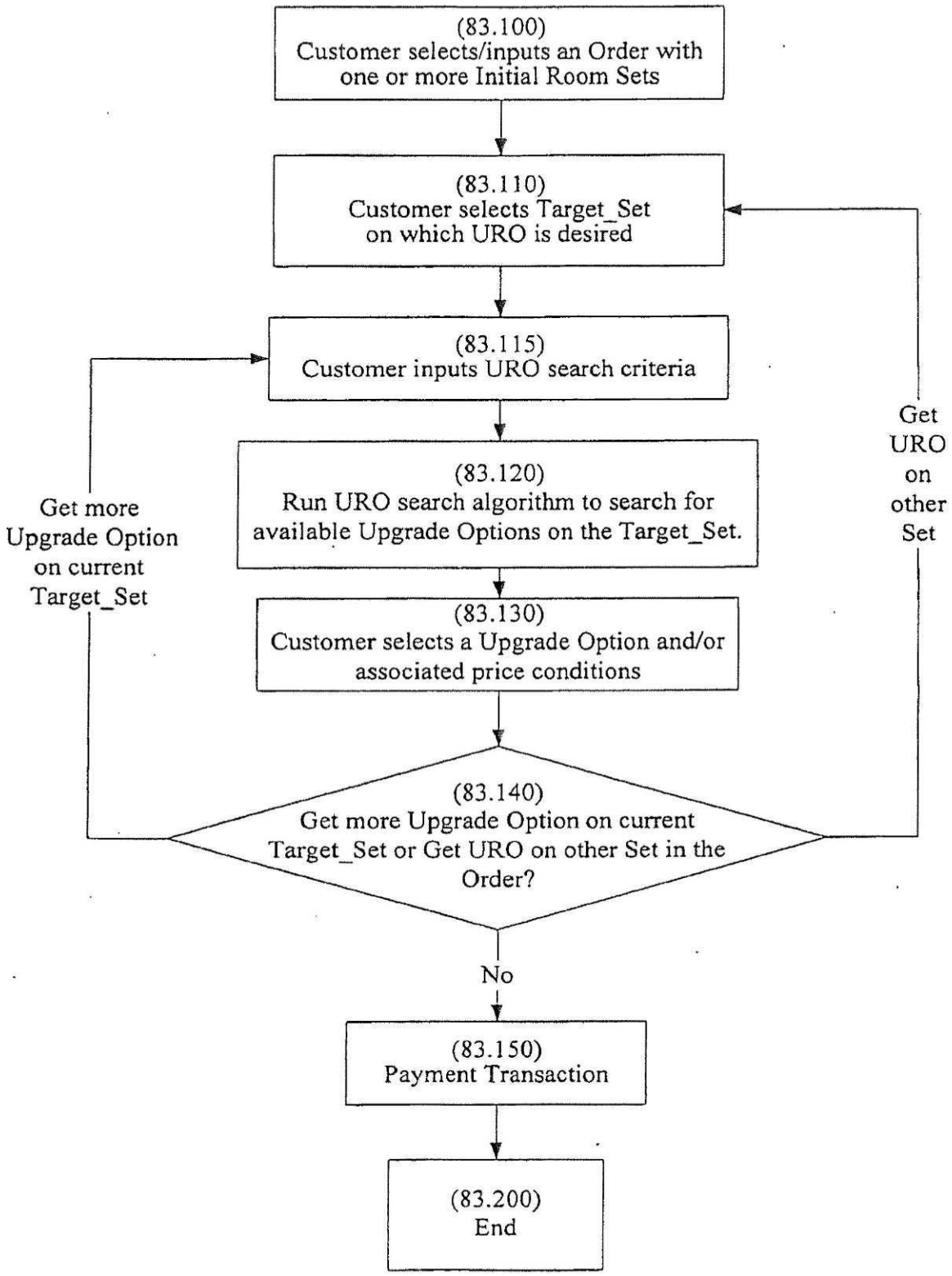

(83.100)
Customer selects/inputs an Order with one or more Initial Room Sets
(83.110)
Customer selects Target_Set on which URO is desired
(83.115)
Customer inputs URO search criteria
(83.120)
Run URO search algorithm to search for available Upgrade Options on the Target_Set.
(83.130)
Customer selects a Upgrade Option and/or associated price conditions
(83.140)
Get more Upgrade Option on current Target_Set or Get URO on other Set in the Order?
Get more Upgrade Option on current Target_Set
Get URO on other Set
No
(83.150)
Payment Transaction
(83.200)
End

Fig. 84

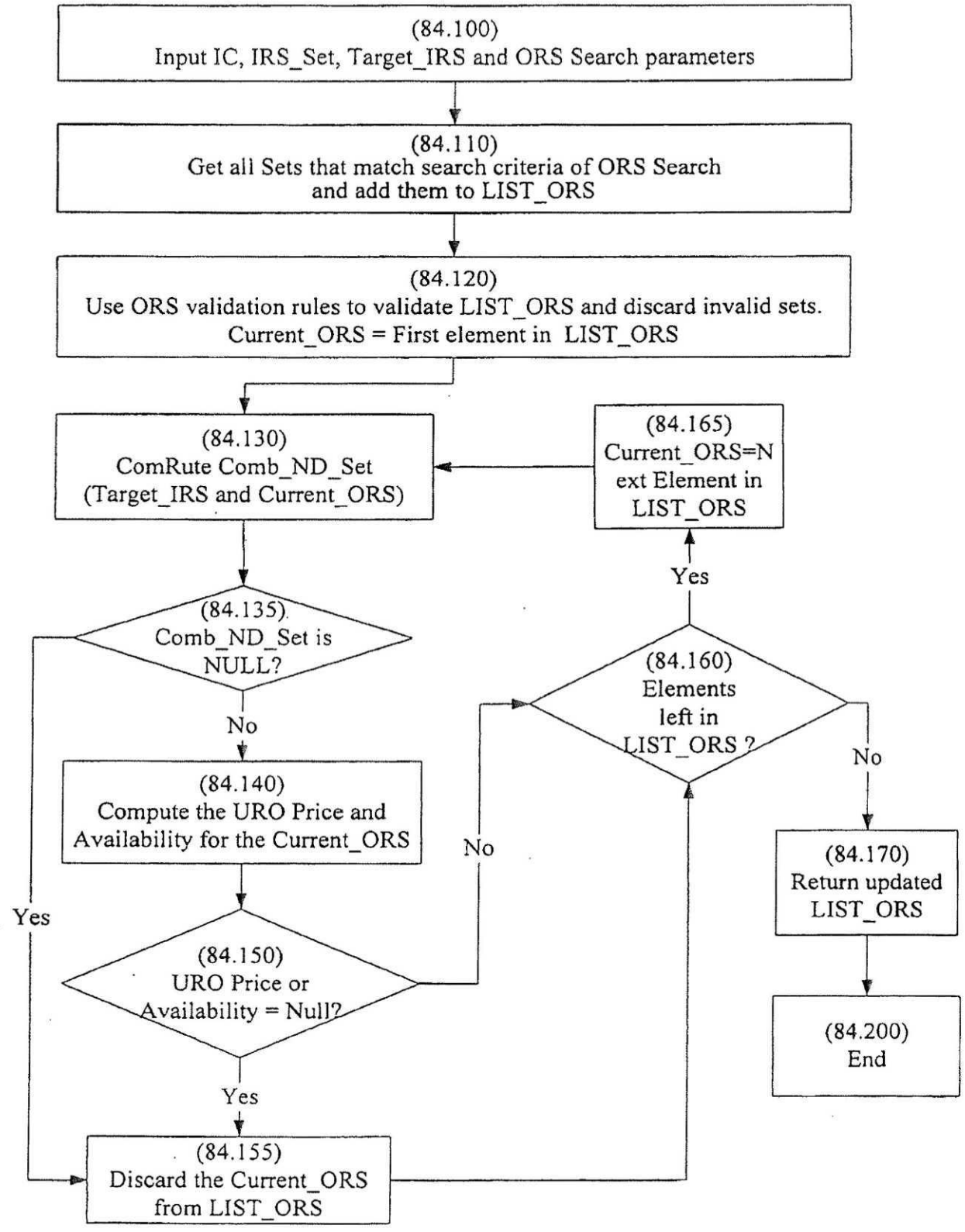

(84.100)
Input IC, IRS_Set, Target_IRS and ORS Search parameters
(84.110)
Get all Sets that match search criteria of ORS Search and add them to LIST_ORS
(84.120)
Use ORS validation rules to validate LIST_ORS and discard invalid sets.
Current_ORS = First element in LIST_ORS
(84.130)
ComRute Comb_ND_Set (Target_IRS and Current_ORS)
(84.165)
Current_ORS=Next Element in LIST_ORS
(84.135)
Comb_ND_Set is NULL?
Yes
No
(84.140)
Compute the URO Price and Availability for the Current_ORS
(84.150)
URO Price or Availability = Null?
Yes
No
(84.155)
Discard the Current_ORS from LIST_ORS
(84.160)
Elements left in LIST_ORS ?
Yes
No
(84.170)
Return updated LIST_ORS
(84.200)
End

Fig. 85

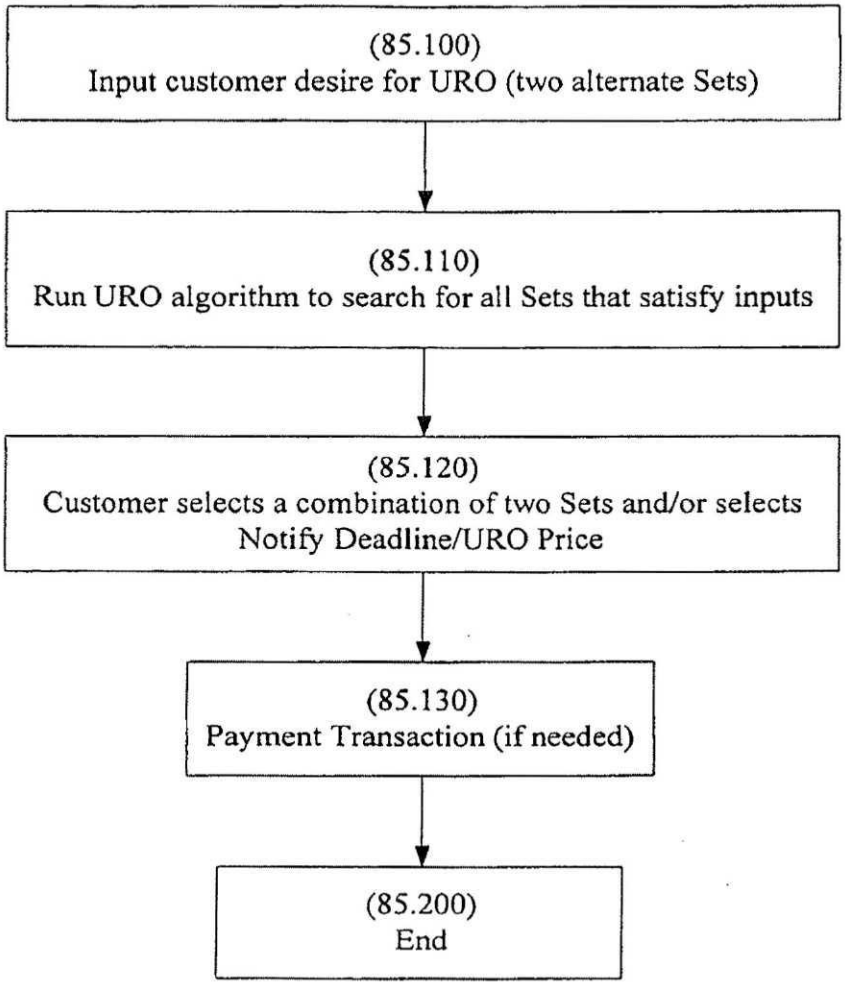

(85.100)
Input customer desire for URO (two alternate Sets)
(85.110)
Run URO algorithm to search for all Sets that satisfy inputs
(85.120)
Customer selects a combination of two Sets and/or selects
Notify Deadline/URO Price
(85.130)
Payment Transaction (if needed)
(85.200)
End

Fig. 86

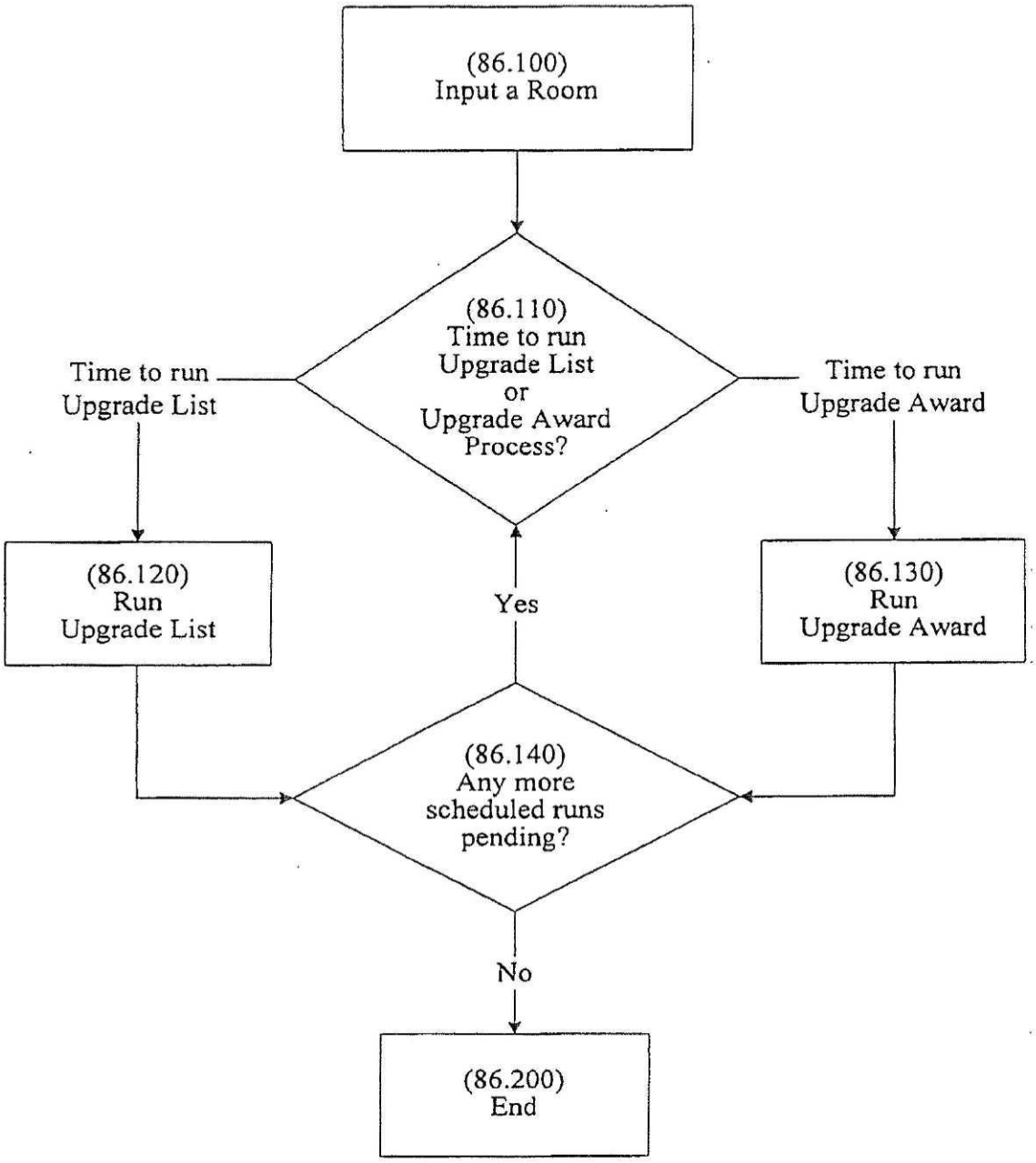

(86.100)
Input a Room
(86.110)
Time to run
Upgrade List
or
Upgrade Award
Process?
Time to run
Upgrade List
Time to run
Upgrade Award
(86.120)
Run
Upgrade List
Yes
(86.130)
Run
Upgrade Award
(86.140)
Any more
scheduled runs
pending?
No
(86.200)
End

Fig. 87

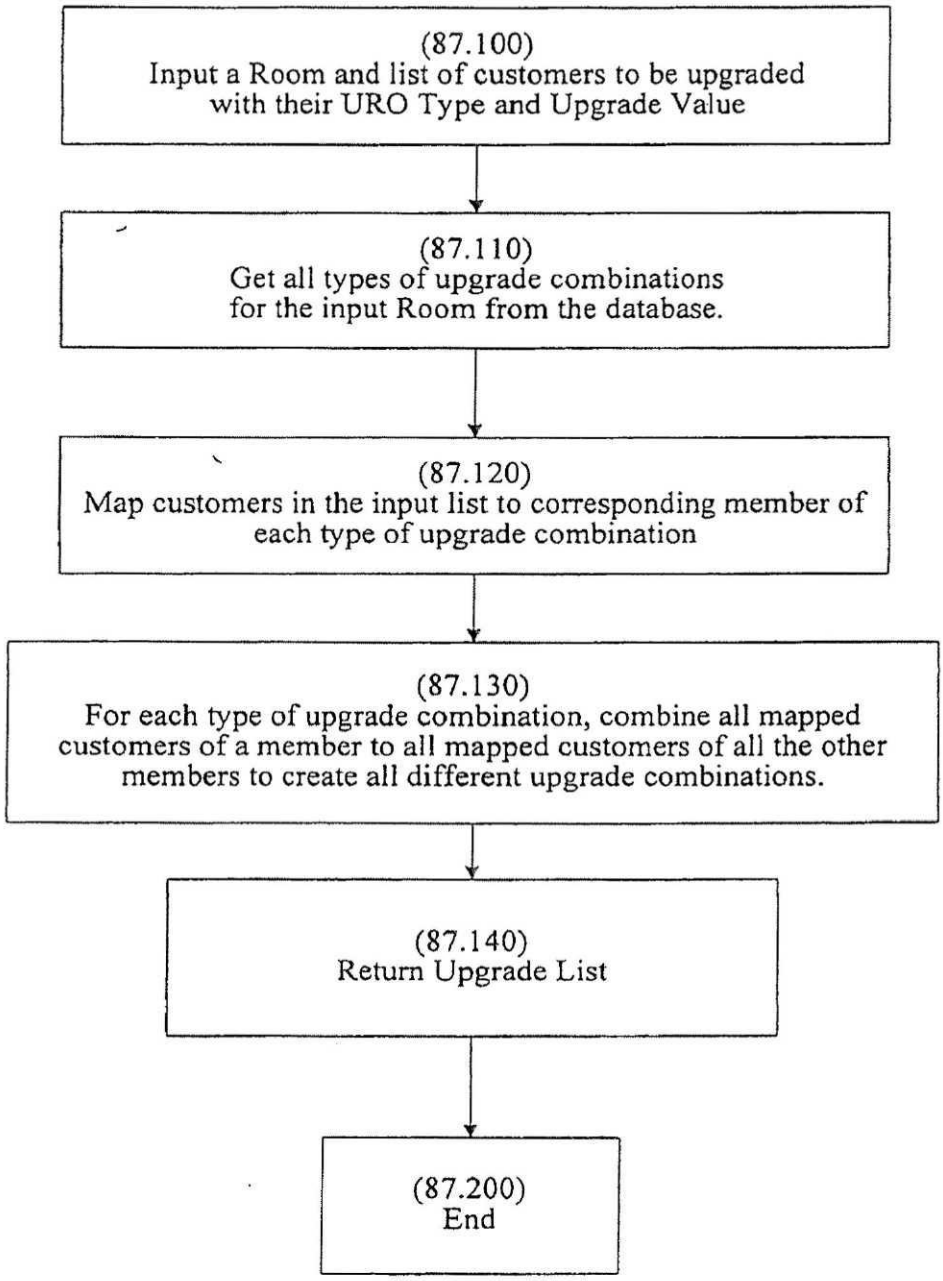

(87.100)
Input a Room and list of customers to be upgraded with their URO Type and Upgrade Value
(87.110)
Get all types of upgrade combinations for the input Room from the database.
(87.120)
Map customers in the input list to corresponding member of each type of upgrade combination
(87.130)
For each type of upgrade combination, combine all mapped customers of a member to all mapped customers of all the other members to create all different upgrade combinations.
(87.140)
Return Upgrade List
(87.200)
End

Fig. 88

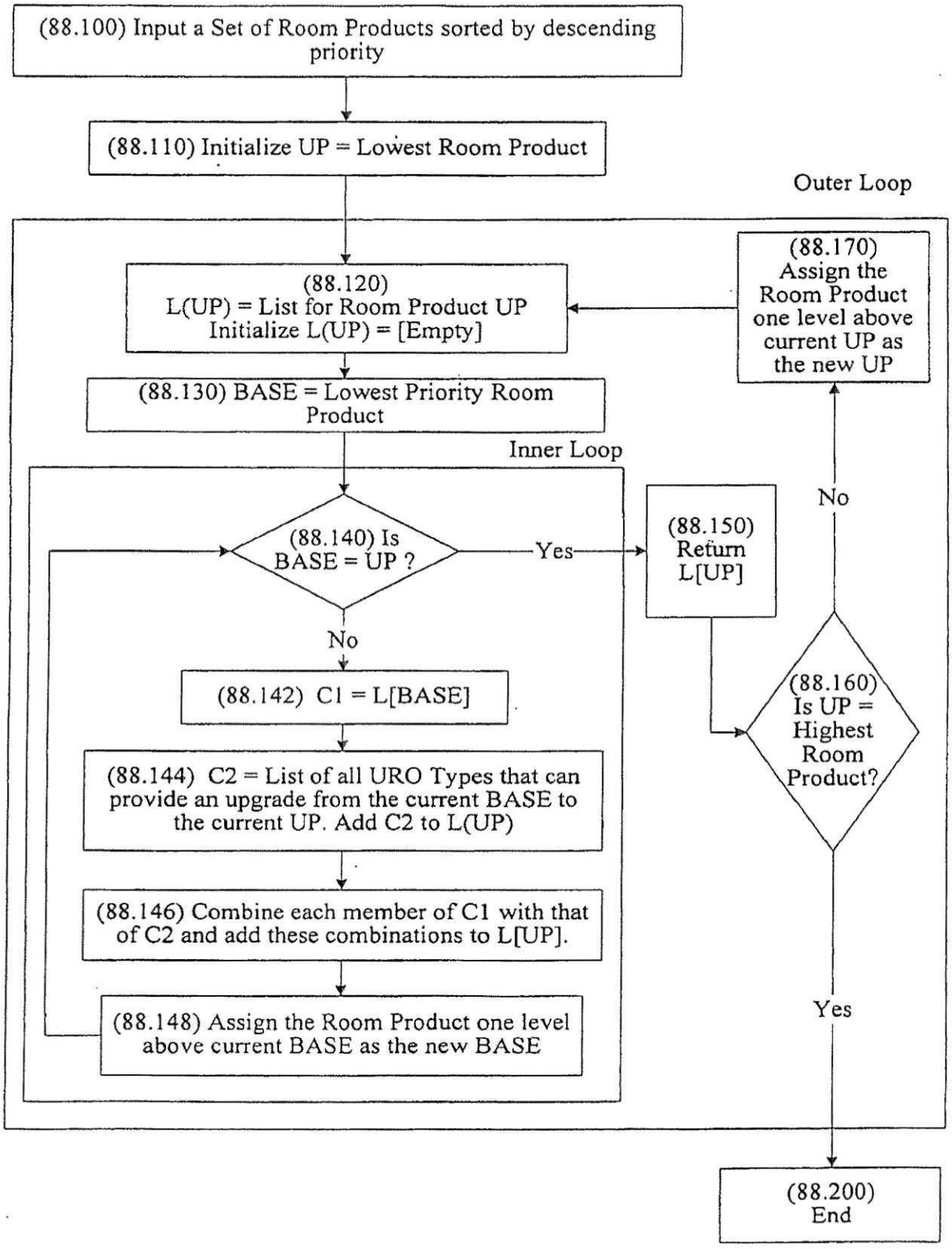

(88.100) Input a Set of Room Products sorted by descending priority
(88.110) Initialize UP = Lowest Room Product
Outer Loop
(88.120) L(UP) = List for Room Product UP Initialize L(UP) = [Empty]
(88.130) BASE = Lowest Priority Room Product
Inner Loop
(88.140) Is BASE = UP ?
Yes
(88.150) Return L[UP]
No
(88.142) C1 = L[BASE]
(88.144) C2 = List of all URO Types that can provide an upgrade from the current BASE to the current UP. Add C2 to L(UP)
(88.146) Combine each member of C1 with that of C2 and add these combinations to L[UP].
(88.148) Assign the Room Product one level above current BASE as the new BASE
(88.160) Is UP = Highest Room Product?
No
(88.170) Assign the Room Product one level above current UP as the new UP
Yes
(88.200) End

Fig. 89

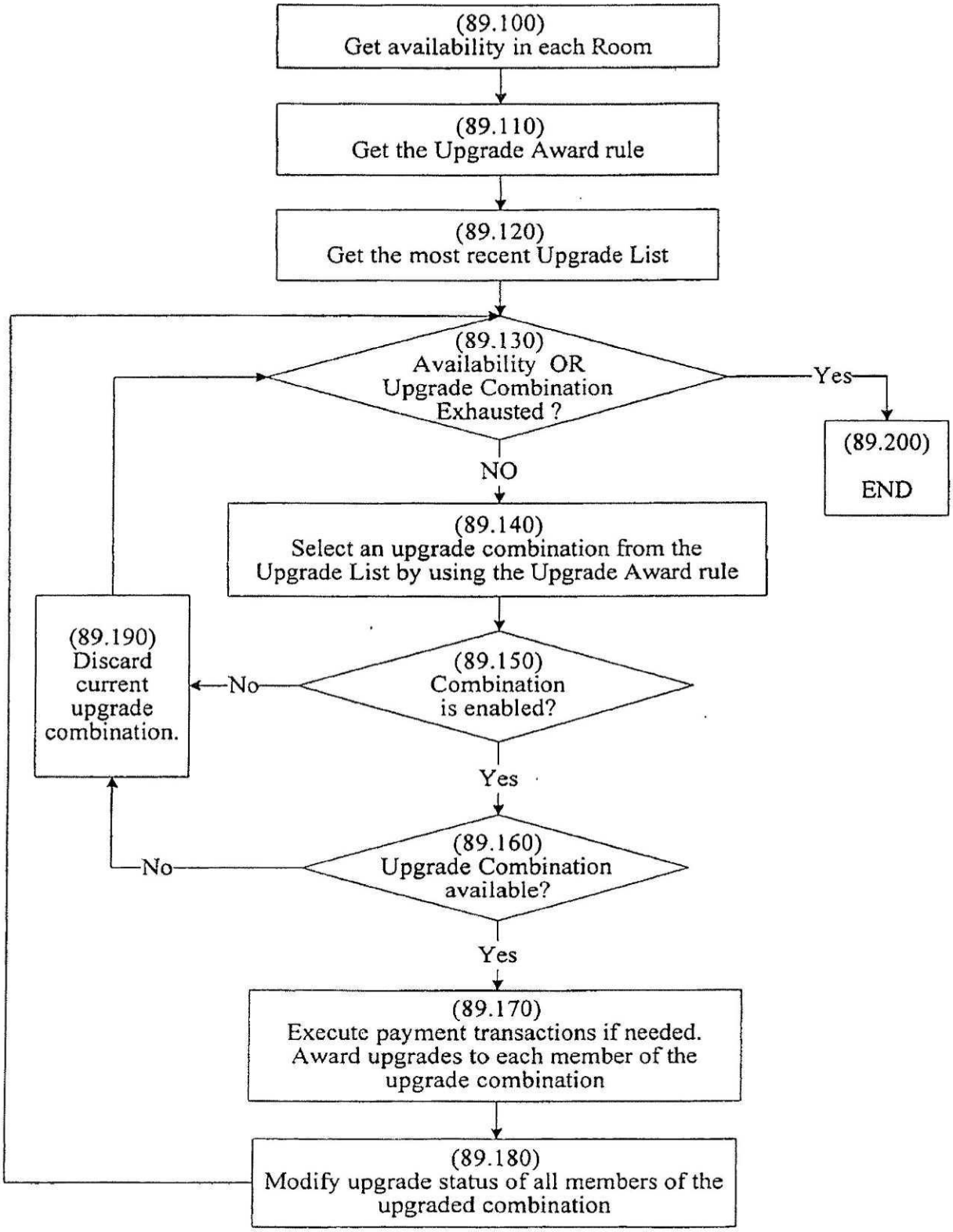

(89.100)
Get availability in each Room
(89.110)
Get the Upgrade Award rule
(89.120)
Get the most recent Upgrade List
(89.130)
Availability OR
Upgrade Combination
Exhausted ?
Yes
(89.200)
END
NO
(89.140)
Select an upgrade combination from the
Upgrade List by using the Upgrade Award rule
(89.150)
Combination
is enabled?
No
(89.190)
Discard
current
upgrade
combination.
Yes
(89.160)
Upgrade Combination
available?
No
Yes
(89.170)
Execute payment transactions if needed.
Award upgrades to each member of the
upgrade combination
(89.180)
Modify upgrade status of all members of the
upgraded combination

Fig. 90

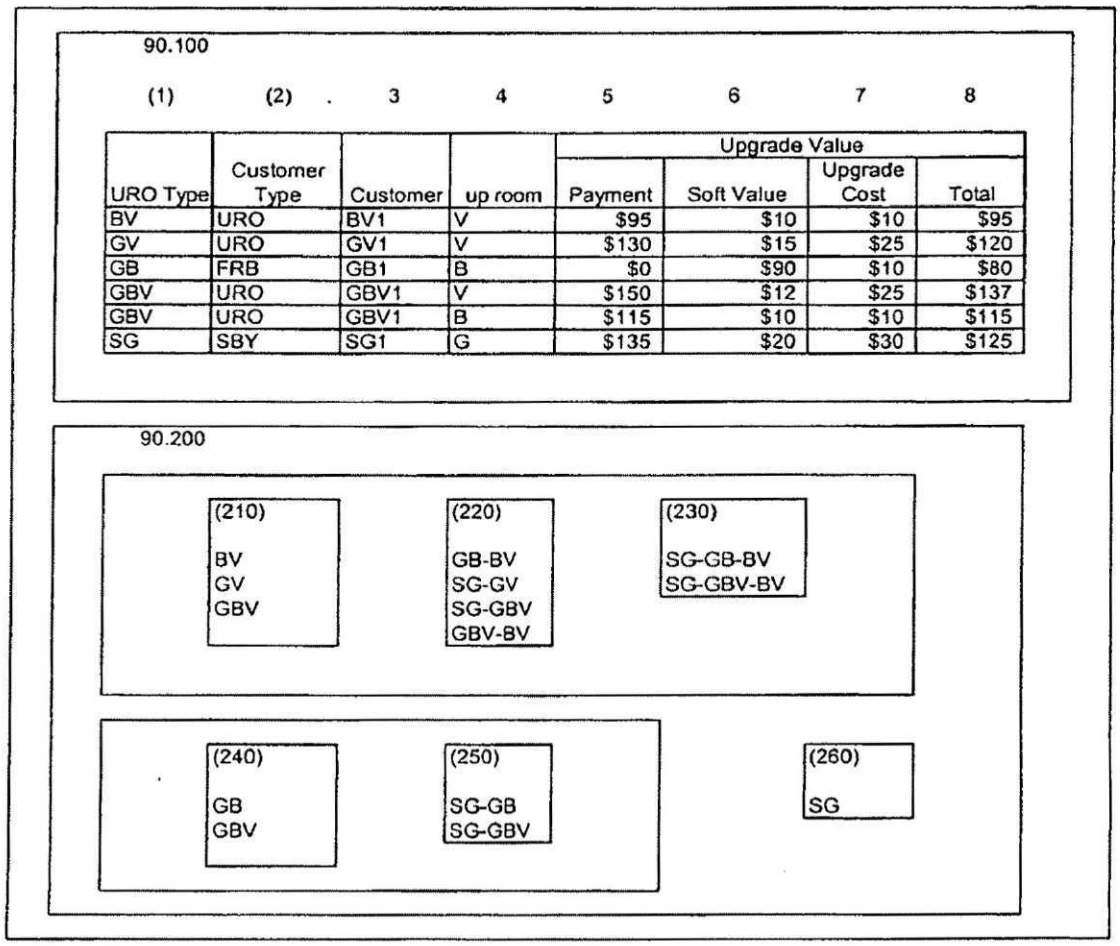

90.100
(1)
(2)
3
4
5
6
7
8
URO Type
Customer Type
Customer
up room
Upgrade Value
Payment
Soft Value
Upgrade Cost
Total
BV URO BV1 V $95 $10 $10 $95
GV URO GV1 V $130 $15 $25 $120
GB FRB GB1 B $0 $90 $10 $80
GBV URO GBV1 V $150 $12 $25 $137
GBV URO GBV1 B $115 $10 $10 $115
SG SBY SG1 G $135 $20 $30 $125
90.200
(210)
BV
GV
GBV
(220)
GB-BV
SG-GV
SG-GBV
GBV-BV
(230)
SG-GB-BV
SG-GBV-BV
(240)
GB
GBV
(250)
SG-GB
SG-GBV
(260)
SG

Fig. 91

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
S. No.	VIP	Business	Guest	Value(V)	Value(B)	Value(G)	Total Value
1	BV1	GBV1	SG1	$95	$115	$125	$335
2	BV1	GB1	SG1	$95	$80	$125	$300
3	GBV1		SG1	$137		$125	$262
4	GV1		SG1	$120		$125	$245
5		*GBV1*	*SG1*		$115	$125	$240
6	BV1		SG1	$95		$125	$220
7	BV1	GBV1		$95	$115		$210
8		*GB1*	*SG1*		$80	$125	$205
9	BV1	GB1		$95	$80		$175
10	GBV1			$137			$137
11			SG1			$125	$125
12	GV1			$120			$120
13		*GBV1*			$115		$115
14	BV1			$95			$95
15		*GB1*			$80		$80

Fig. 92

(92.100)

	VIP	Business	Guest
Availability	1	1	0

(92.200)

Row No	S. No.	VIP	Business	Guest	Total Value
1	1	BV1	GBV1	SG1	$335
15	2		GB1		$80
			Total Upgrade Value		$415

Fig. 93

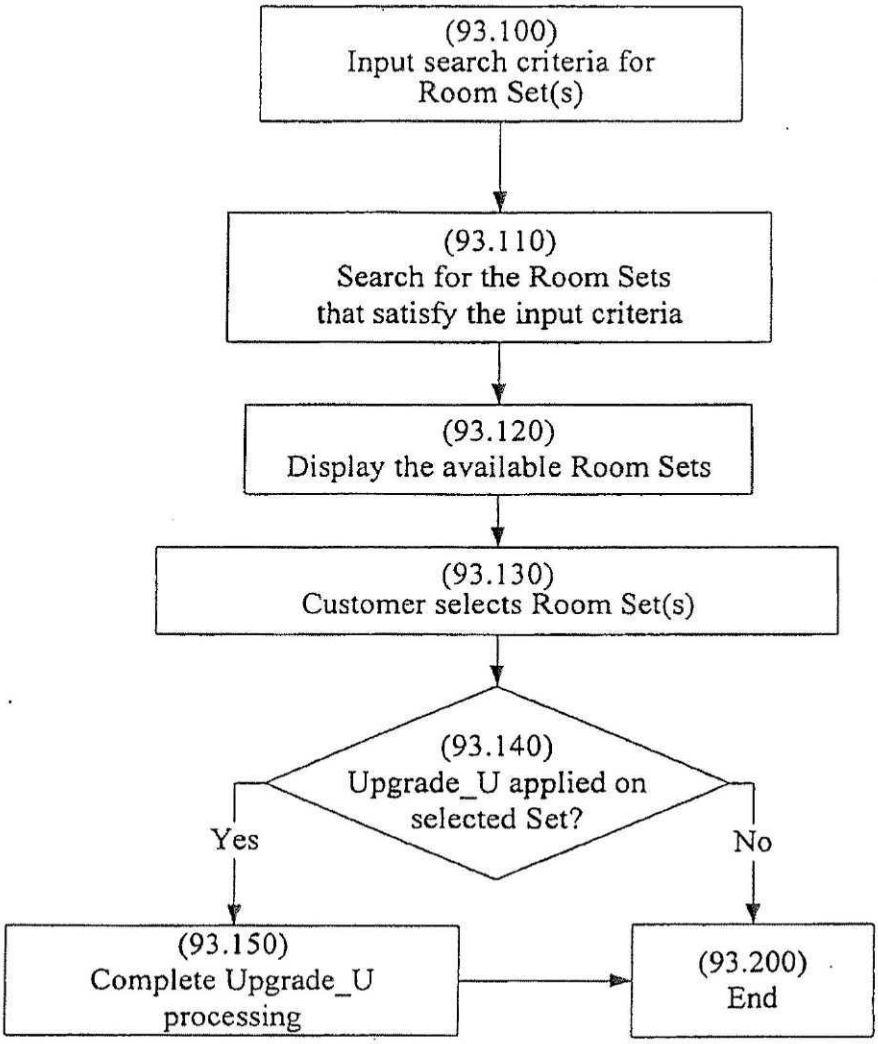

(93.100)
Input search criteria for Room Set(s)
(93.110)
Search for the Room Sets that satisfy the input criteria
(93.120)
Display the available Room Sets
(93.130)
Customer selects Room Set(s)
(93.140)
Upgrade_U applied on selected Set?
Yes
No
(93.150)
Complete Upgrade_U processing
(93.200)
End

Fig. 94

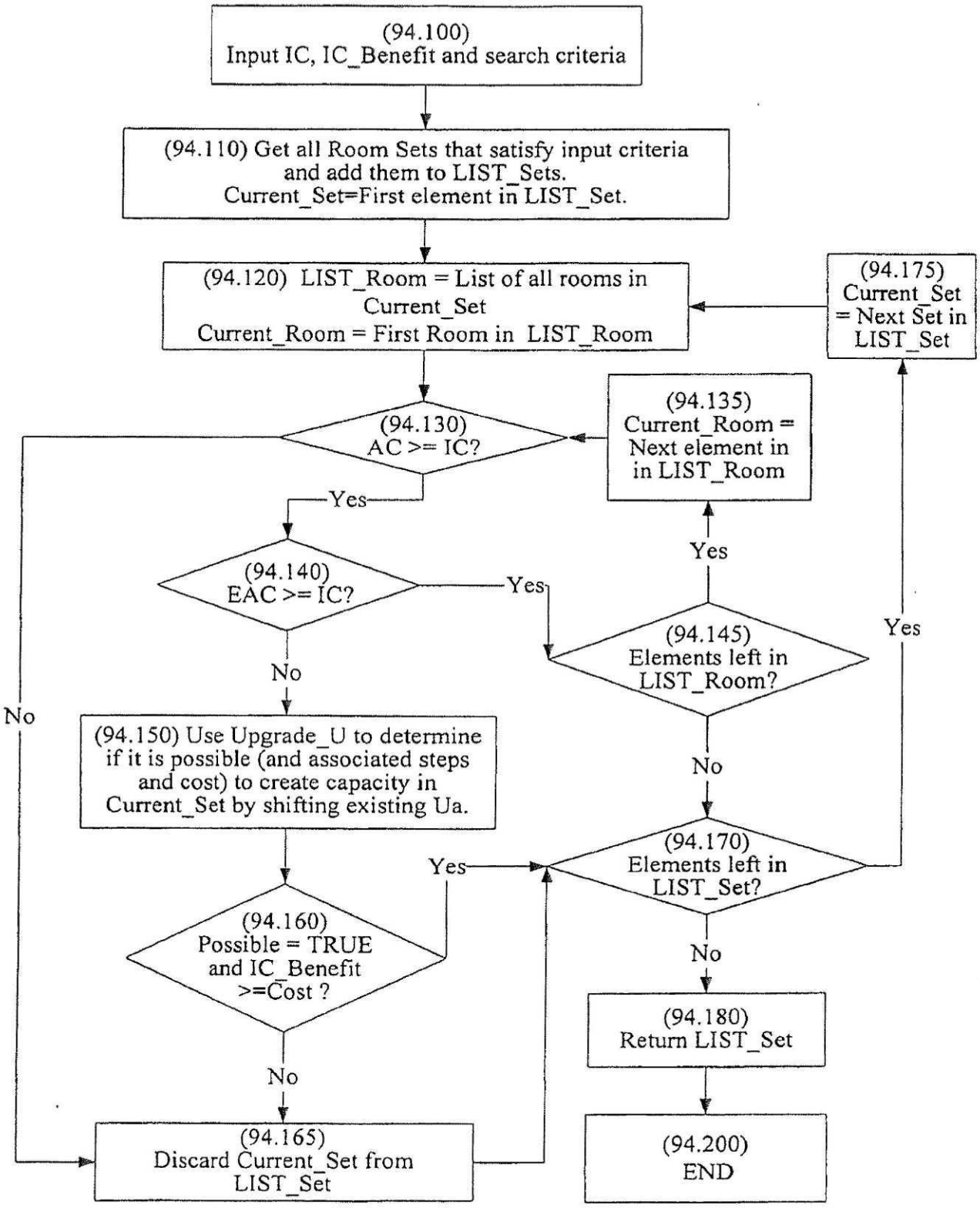

(94.100)
Input IC, IC_Benefit and search criteria
(94.110) Get all Room Sets that satisfy input criteria and add them to LIST_Sets.
Current_Set=First element in LIST_Set.
(94.120) LIST_Room = List of all rooms in Current_Set
Current_Room = First Room in _LIST_Room
(94.175) Current_Set = Next Set in LIST_Set
(94.130) AC >= IC?
(94.135) Current_Room = Next element in in LIST_Room
Yes
No
(94.140) EAC >= IC?
Yes
No
Yes
(94.145) Elements left in LIST_Room?
Yes
No
(94.150) Use Upgrade_U to determine if it is possible (and associated steps and cost) to create capacity in Current_Set by shifting existing Ua.
(94.170) Elements left in LIST_Set?
Yes
No
(94.160) Possible = TRUE and IC_Benefit >=Cost ?
No
(94.180) Return LIST_Set
(94.165) Discard Current_Set from LIST_Set
(94.200) END

Fig. 95

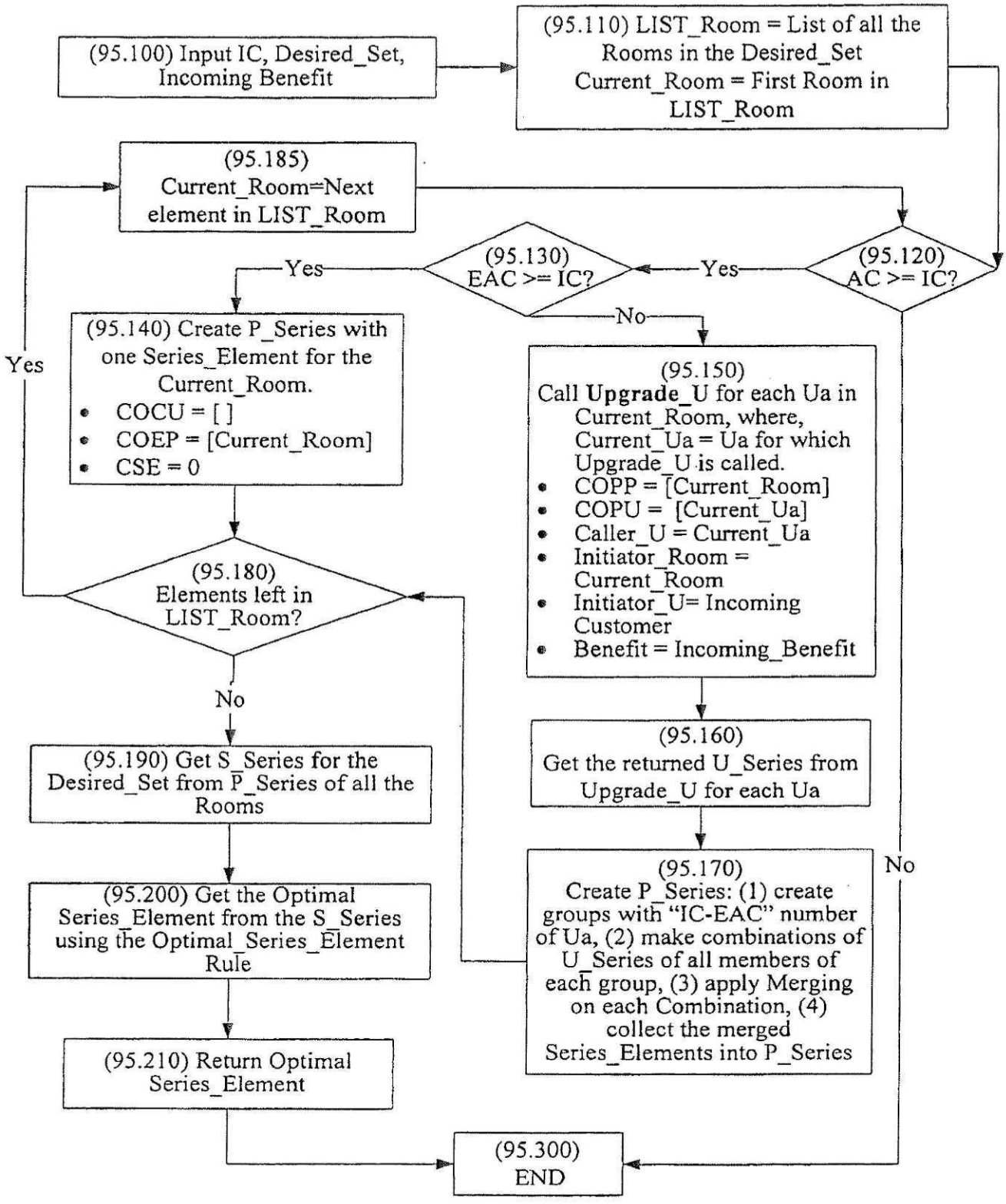

(95.100) Input IC, Desired_Set, Incoming Benefit
(95.110) LIST_Room = List of all the Rooms in the Desired_Set Current_Room = First Room in LIST_Room
(95.185) Current_Room=Next element in LIST_Room
(95.120) AC >= IC?
Yes
(95.130) EAC >= IC?
Yes
No
(95.140) Create P_Series with one Series_Element for the Current_Room.
COCU = [ ]
COEP = [Current_Room]
CSE = 0
(95.150) Call Upgrade_U for each Ua in Current_Room, where, Current_Ua = Ua for which Upgrade_U is called.
COPP = [Current_Room]
COPU = [Current_Ua]
Caller_U = Current_Ua
Initiator_Room = Current_Room
Initiator_U= Incoming Customer
Benefit = Incoming_Benefit
Yes
(95.180) Elements left in LIST_Room?
No
(95.160) Get the returned U_Series from Upgrade_U for each Ua
(95.190) Get S_Series for the Desired_Set from P_Series of all the Rooms
(95.170) Create P_Series: (1) create groups with "IC-EAC" number of Ua, (2) make combinations of U_Series of all members of each group, (3) apply Merging on each Combination, (4) collect the merged Series_Elements into P_Series
No
(95.200) Get the Optimal Series_Element from the S_Series using the Optimal_Series_Element Rule
(95.210) Return Optimal Series_Element
(95.300) END

Fig. 96

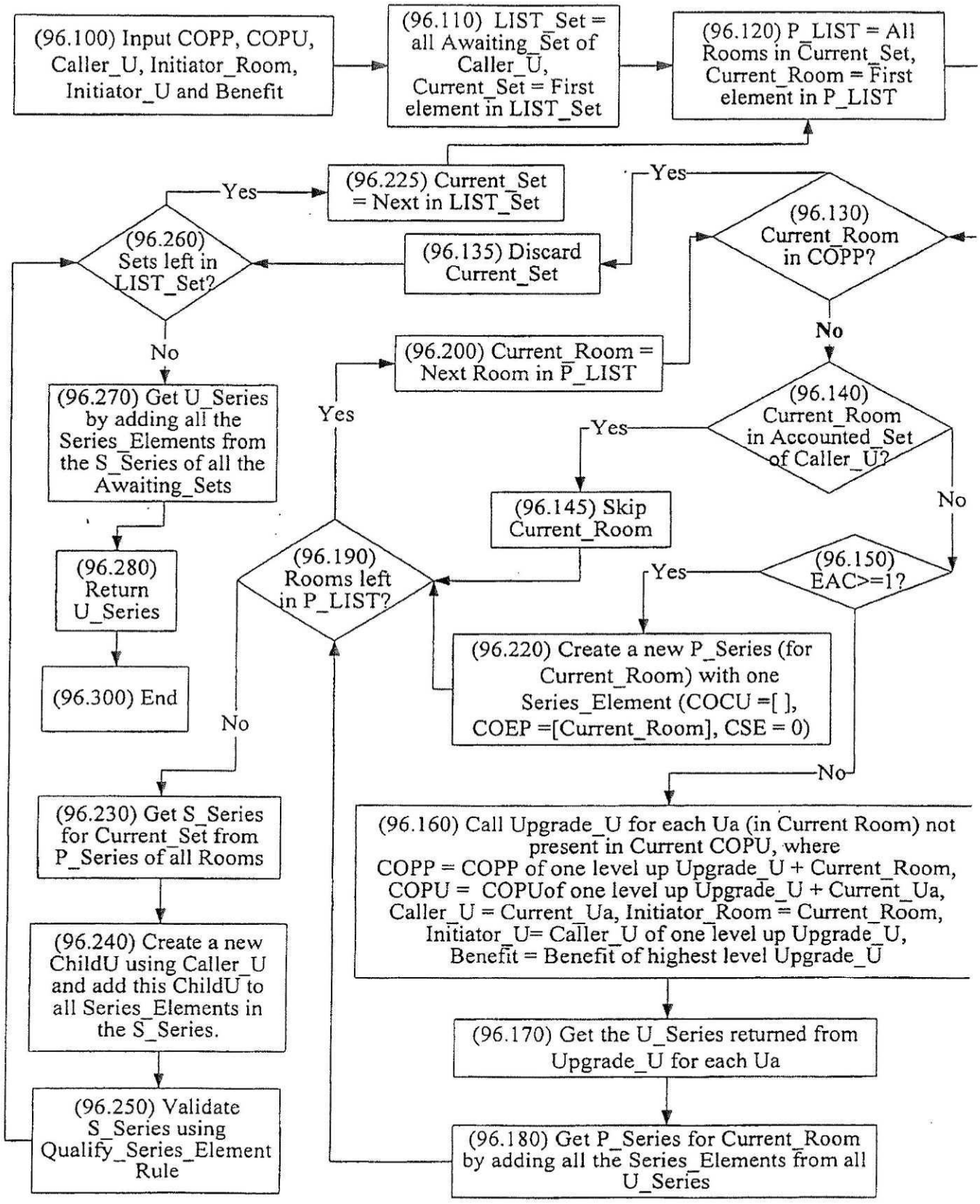

(96.100) Input COPP, COPU, Caller_U, Initiator_Room, Initiator_U and Benefit
(96.110) LIST_Set = all Awaiting_Set of Caller_U, Current_Set = First element in LIST_Set
(96.120) P_LIST = All Rooms in Current_Set, Current_Room = First element in P_LIST
(96.225) Current_Set = Next in LIST_Set
Yes
(96.260) Sets left in LIST_Set?
(96.135) Discard Current_Set
Yes
(96.130) Current_Room in COPP?
No
(96.200) Current_Room = Next Room in P_LIST
Yes
(96.140) Current_Room in Accounted_Set of Caller_U?
Yes
(96.145) Skip Current_Room
No
(96.150) EAC>=1?
Yes
(96.220) Create a new P_Series (for Current_Room) with one Series_Element (COCU =[ ], COEP =[Current_Room], CSE = 0)
No
(96.160) Call Upgrade_U for each Ua (in Current Room) not present in Current COPU, where COPP = COPP of one level up Upgrade_U + Current_Room, COPU = COPUof one level up Upgrade_U + Current_Ua, Caller_U = Current_Ua, Initiator_Room = Current_Room, Initiator_U= Caller_U of one level up Upgrade_U, Benefit = Benefit of highest level Upgrade_U
(96.170) Get the U_Series returned from Upgrade_U for each Ua
(96.180) Get P_Series for Current_Room by adding all the Series_Elements from all U_Series
(96.190) Rooms left in P_LIST?
Yes
No
(96.270) Get U_Series by adding all the Series_Elements from the S_Series of all the Awaiting_Sets
(96.280) Return U_Series
(96.300) End
No
(96.230) Get S_Series for Current_Set from P_Series of all Rooms
(96.240) Create a new ChildU using Caller_U and add this ChildU to all Series_Elements in the S_Series.
(96.250) Validate S_Series using Qualify_Series_Element Rule

Fig. 97

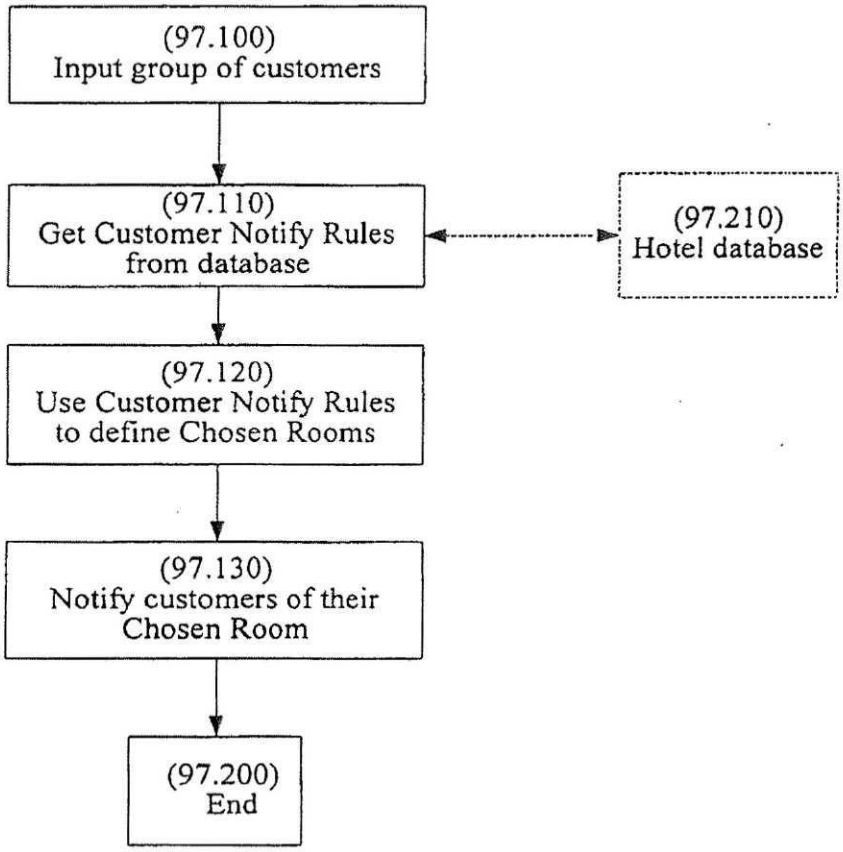

(97.100)
Input group of customers
(97.110)
Get Customer Notify Rules
from database
(97.210)
Hotel database
(97.120)
Use Customer Notify Rules
to define Chosen Rooms
(97.130)
Notify customers of their
Chosen Room
(97.200)
End

Fig. 98

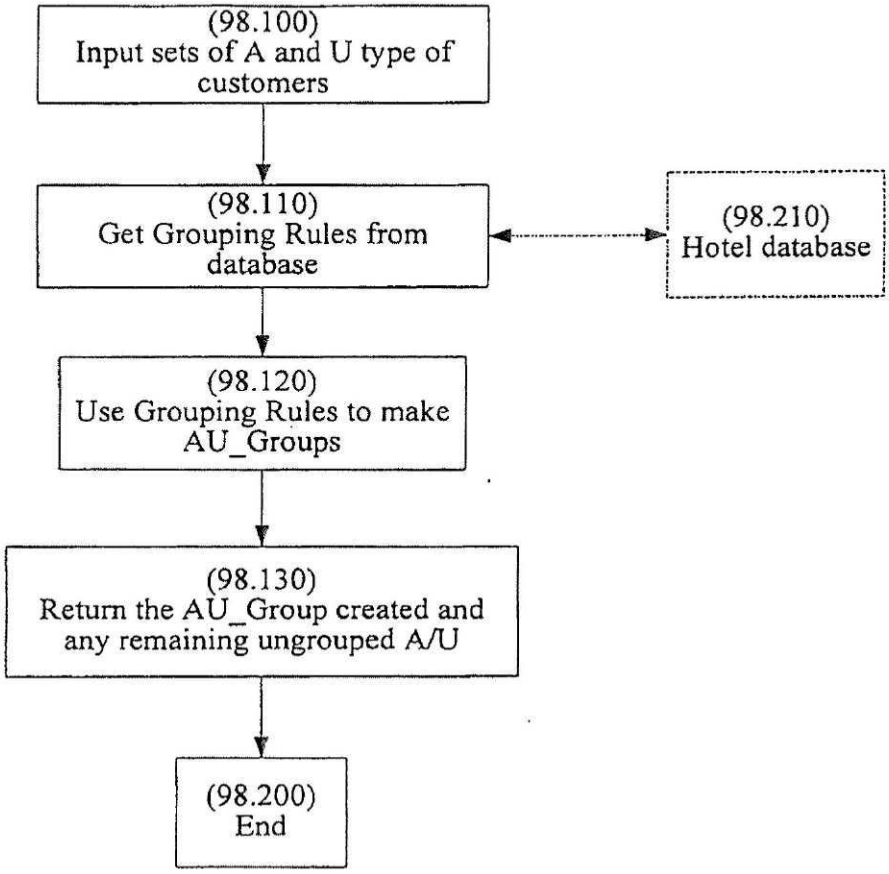

(98.100)
Input sets of A and U type of customers
(98.110)
Get Grouping Rules from database
(98.210)
Hotel database
(98.120)
Use Grouping Rules to make AU_Groups
(98.130)
Return the AU_Group created and any remaining ungrouped A/U
(98.200)
End

Fig. 99

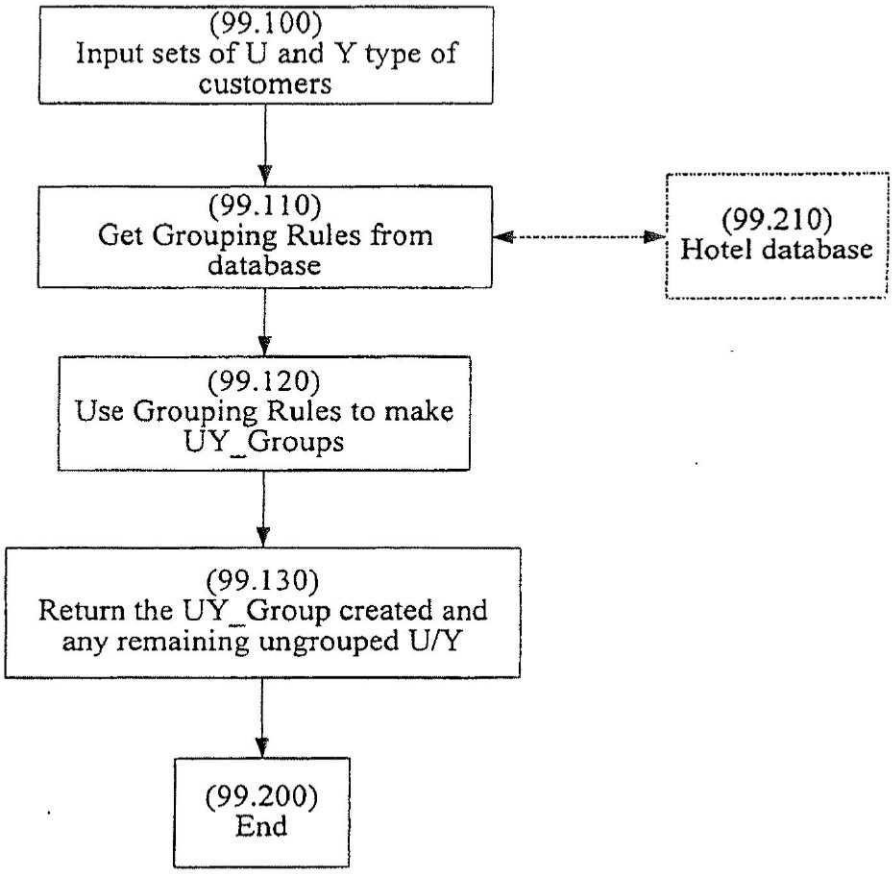

(99.100)
Input sets of U and Y type of customers
(99.110)
Get Grouping Rules from database
(99.210)
Hotel database
(99.120)
Use Grouping Rules to make UY_Groups
(99.130)
Return the UY_Group created and any remaining ungrouped U/Y
(99.200)
End

Fig. 100

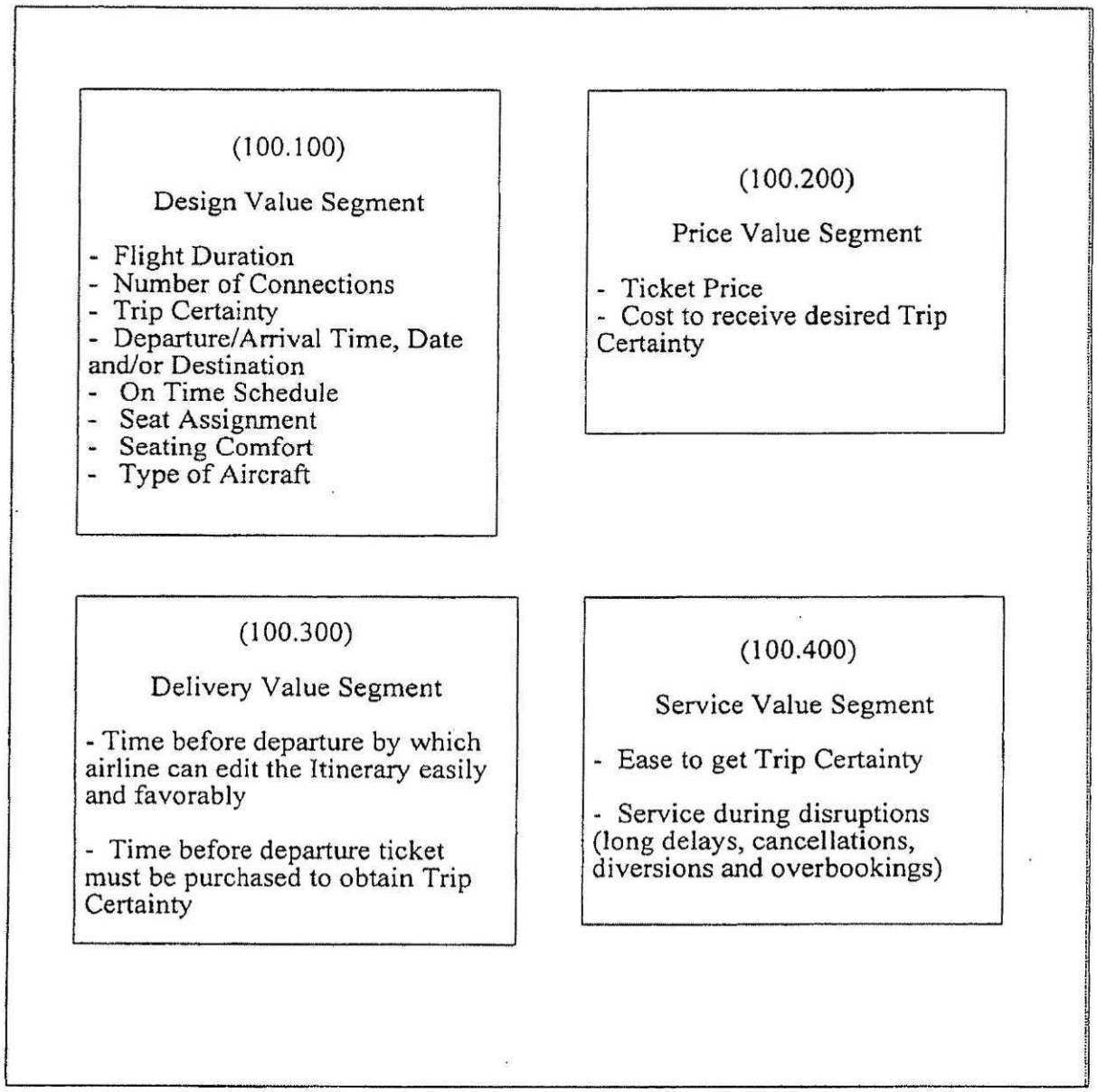

(100.100)
Design Value Segment
- Flight Duration
- Number of Connections
- Trip Certainty
- Departure/Arrival Time, Date and/or Destination
- On Time Schedule
- Seat Assignment
- Seating Comfort
- Type of Aircraft
(100.200)
Price Value Segment
- Ticket Price
- Cost to receive desired Trip Certainty
(100.300)
Delivery Value Segment
- Time before departure by which airline can edit the Itinerary easily and favorably
- Time before departure ticket must be purchased to obtain Trip Certainty
(100.400)
Service Value Segment
- Ease to get Trip Certainty
- Service during disruptions (long delays, cancellations, diversions and overbookings)

Fig. 101

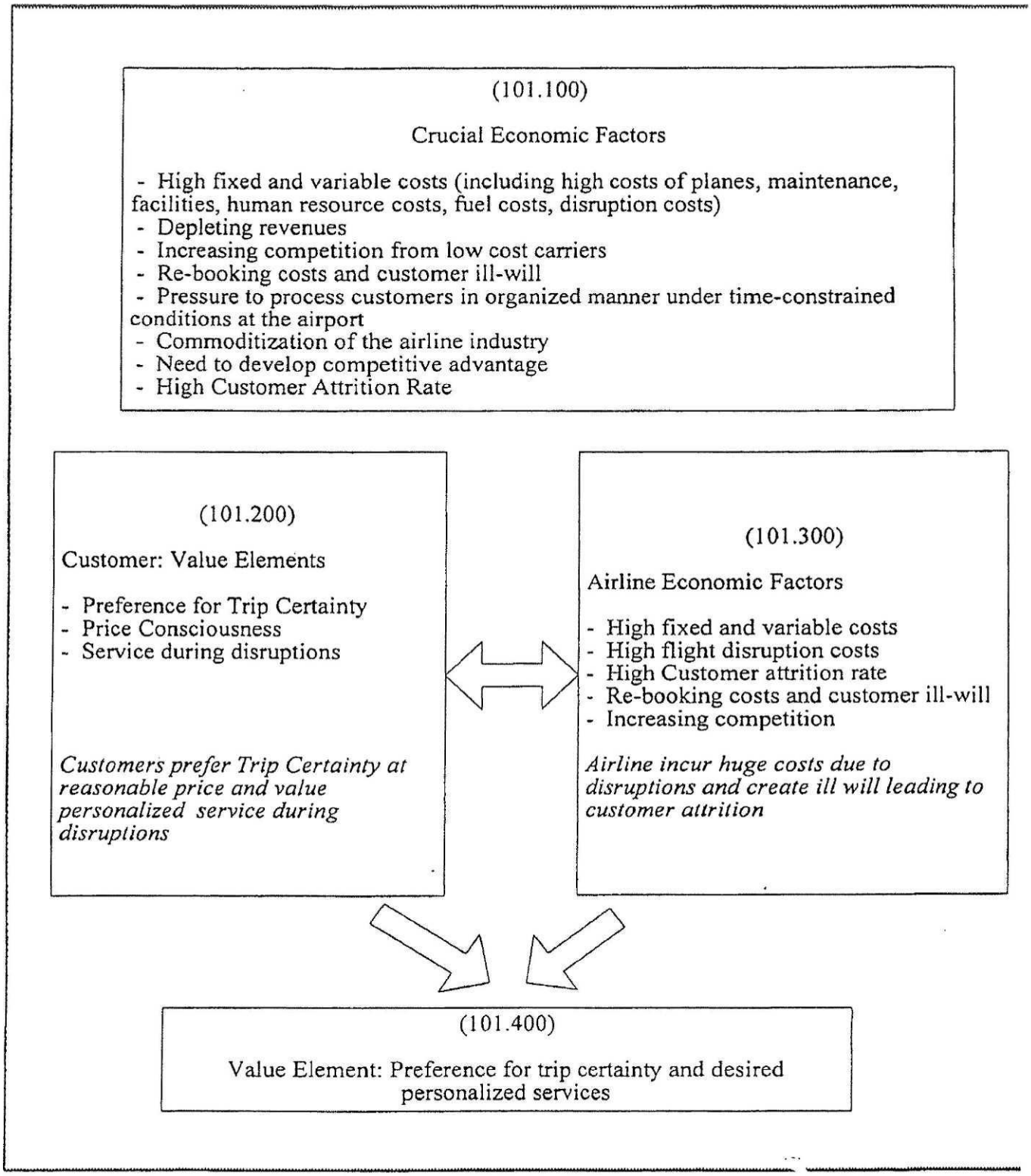

(101.100)
Crucial Economic Factors
- High fixed and variable costs (including high costs of planes, maintenance, facilities, human resource costs, fuel costs, disruption costs)
- Depleting revenues
- Increasing competition from low cost carriers
- Re-booking costs and customer ill-will
- Pressure to process customers in organized manner under time-constrained conditions at the airport
- Commoditization of the airline industry
- Need to develop competitive advantage
- High Customer Attrition Rate
(101.200)
Customer: Value Elements
- Preference for Trip Certainty
- Price Consciousness
- Service during disruptions
Customers prefer Trip Certainty at reasonable price and value personalized service during disruptions
(101.300)
Airline Economic Factors
- High fixed and variable costs
- High flight disruption costs
- High Customer attrition rate
- Re-booking costs and customer ill-will
- Increasing competition
Airline incur huge costs due to disruptions and create ill will leading to customer attrition
(101.400)
Value Element: Preference for trip certainty and desired personalized services

Fig. 102

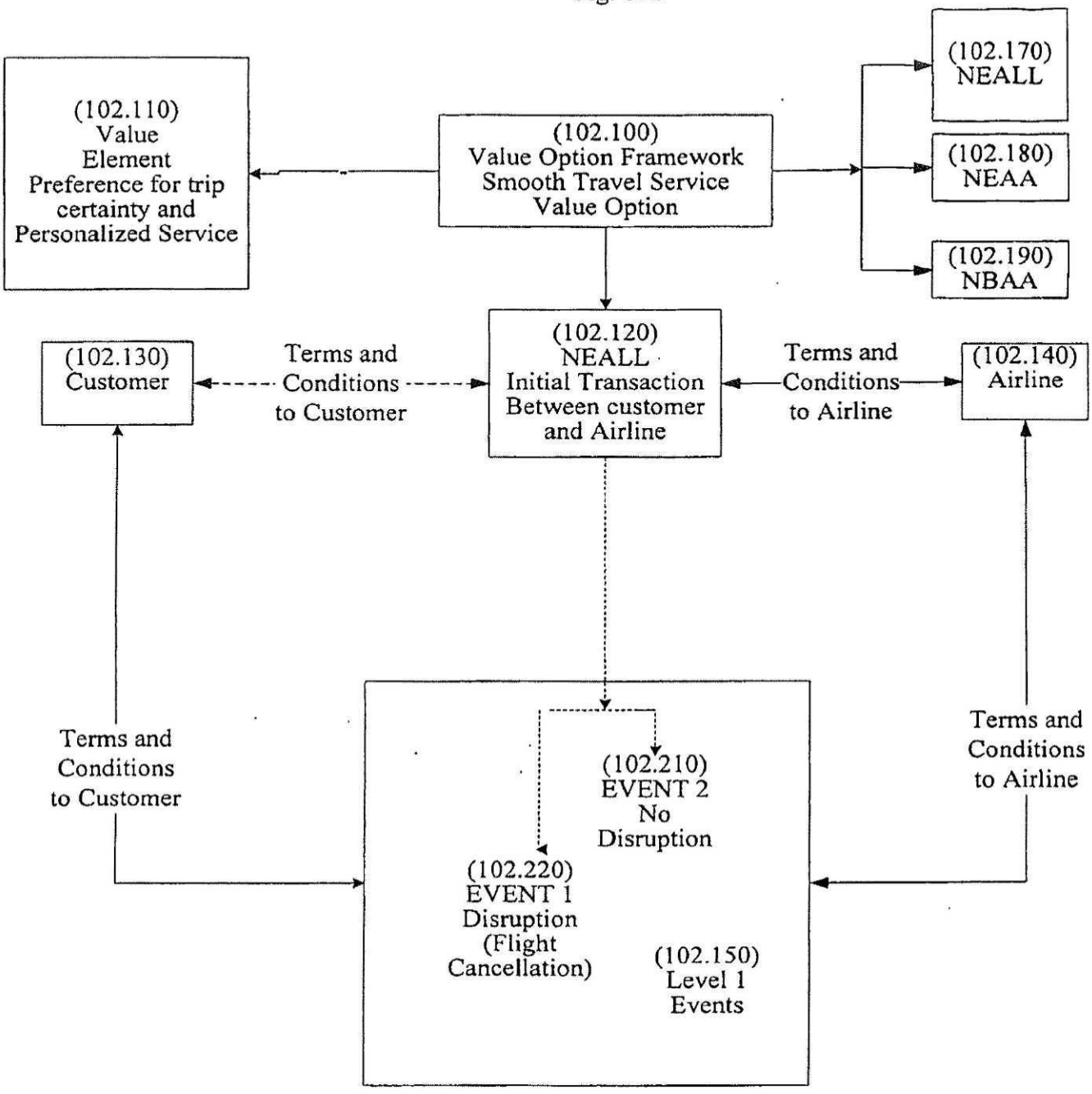

(102.110)
Value
Element
Preference for trip
certainty and
Personalized Service
(102.100)
Value Option Framework
Smooth Travel Service
Value Option
(102.170)
NEALL
(102.180)
NEAA
(102.190)
NBAA
(102.120)
NEALL
Initial Transaction
Between customer
and Airline
(102.130)
Customer
Terms and
Conditions
to Customer
Terms and
Conditions
to Airline
(102.140)
Airline
Terms and
Conditions
to Customer
(102.210)
EVENT 2
No
Disruption
(102.220)
EVENT 1
Disruption
(Flight
Cancellation)
(102.150)
Level 1
Events
Terms and
Conditions
to Airline

Fig. 103

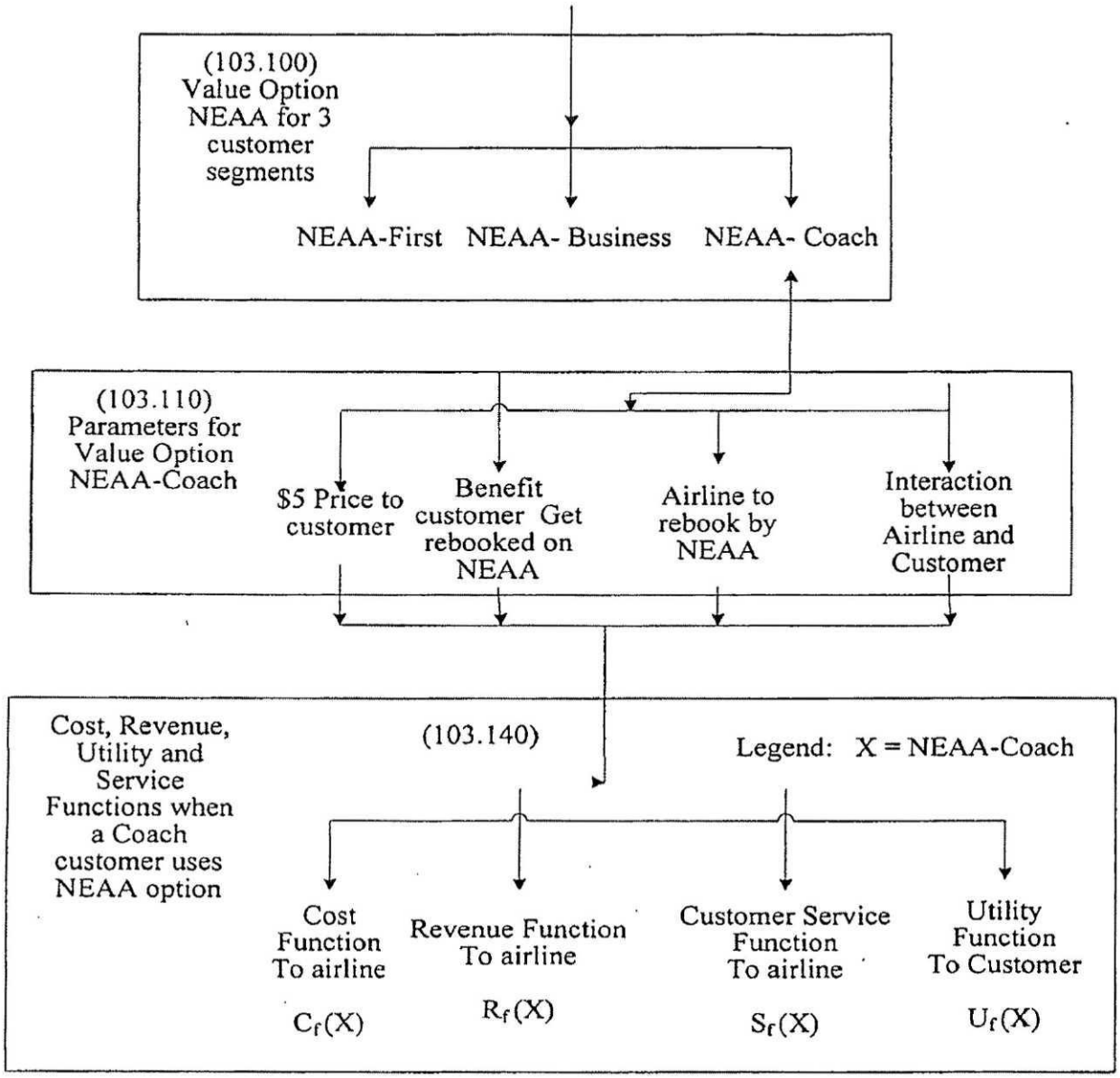
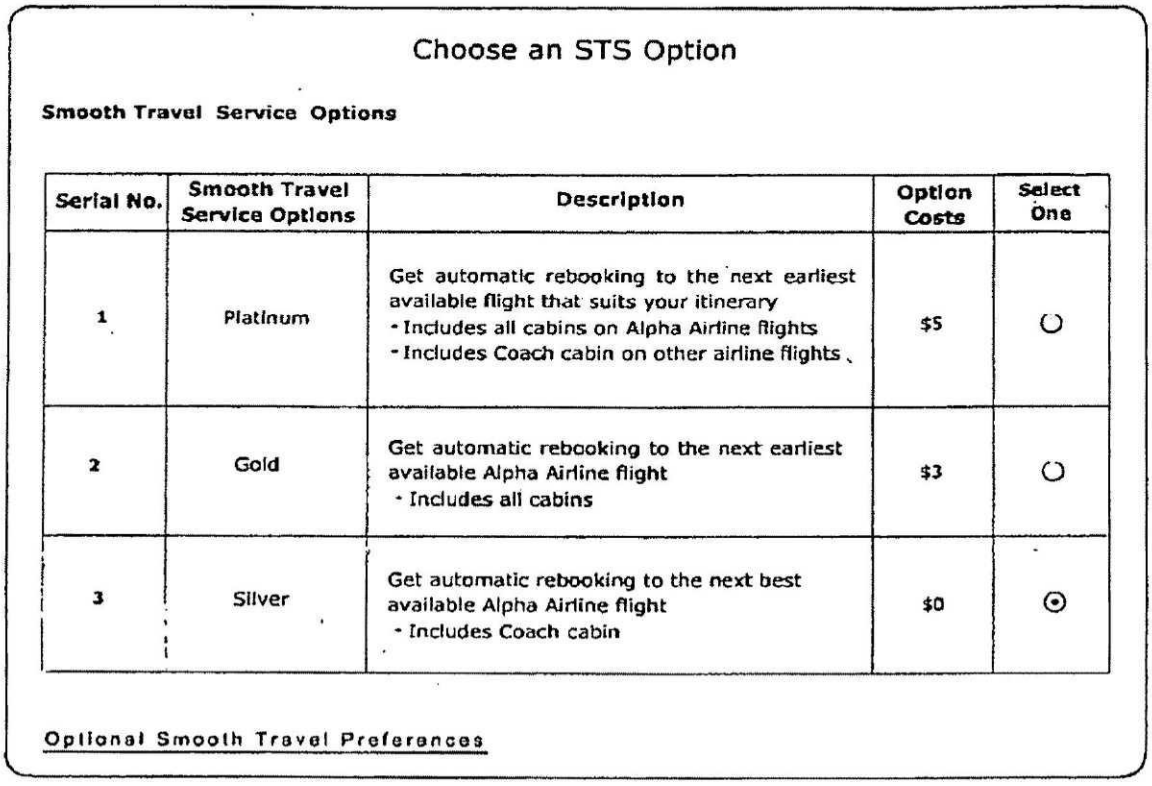

(103.100)
Value Option
NEAA for 3
customer
segments
NEAA-First
NEAA- Business
NEAA- Coach
(103.110)
Parameters for
Value Option
NEAA-Coach
$5 Price to
customer
Benefit
customer Get
rebooked on
NEAA
Airline to
rebook by
NEAA
Interaction
between
Airline and
Customer
Cost, Revenue,
Utility and
Service
Functions when
a Coach
customer uses
NEAA option
(103.140)
Legend: X = NEAA-Coach
Cost
Function
To airline
$C_f(X)$
Revenue Function
To airline
$R_f(X)$
Customer Service
Function
To airline
$S_f(X)$
Utility
Function
To Customer
$U_f(X)$

Fig. 104



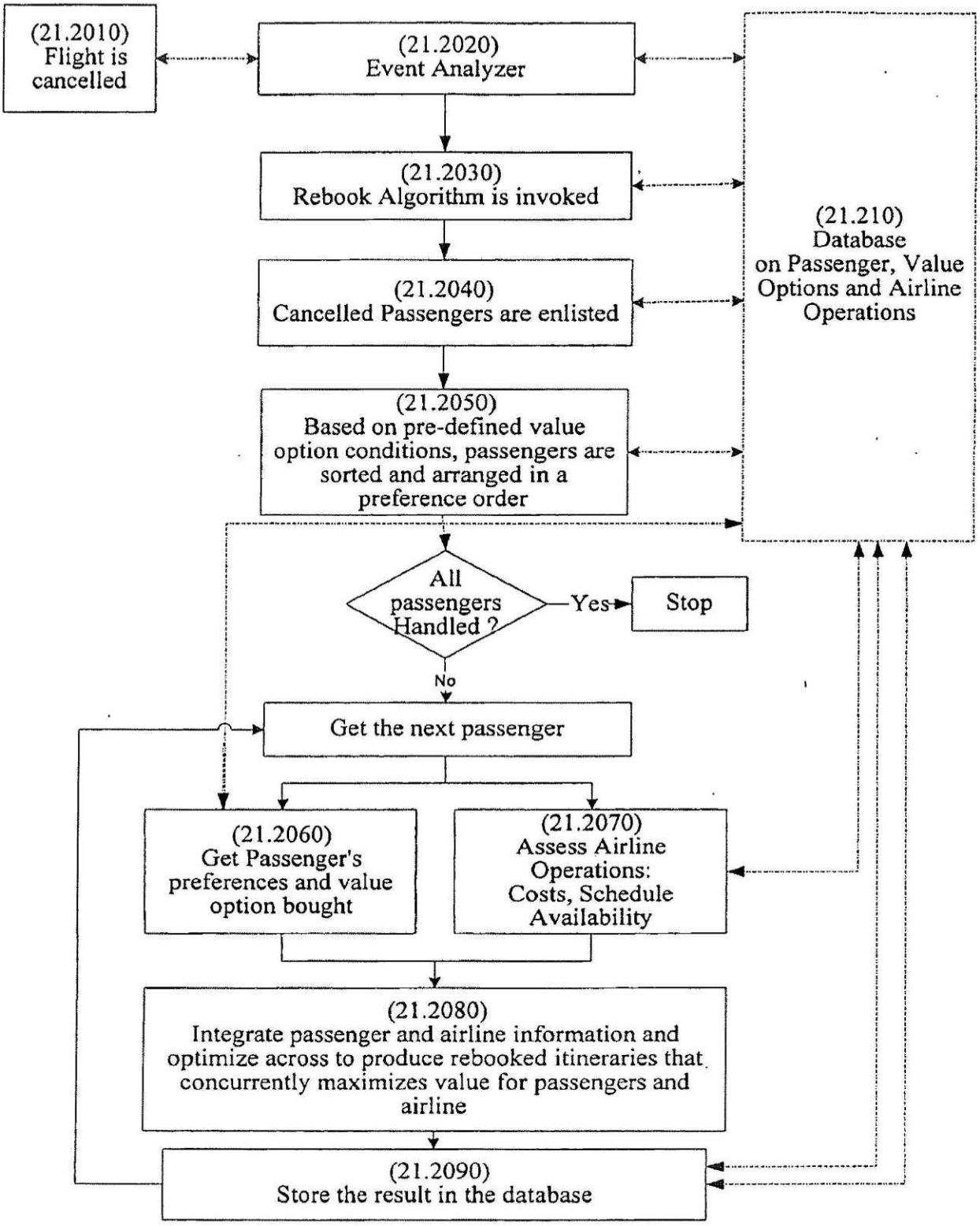
Choose an STS Option

Smooth Travel Service Options

Serial No.	Smooth Travel Service Options	Description	Option Costs	Select One
1	Platinum	Get automatic rebooking to the next earliest available flight that suits your itinerary - Includes all cabins on Alpha Airline flights - Includes Coach cabin on other airline flights	$5	○
2	Gold	Get automatic rebooking to the next earliest available Alpha Airline flight - Includes all cabins	$3	○
3	Silver	Get automatic rebooking to the next best available Alpha Airline flight - Includes Coach cabin	$0	⊙

Optional Smooth Travel Preferences

Fig. 105


(21.2010)
Flight is cancelled
(21.2020)
Event Analyzer
(21.2030)
Rebook Algorithm is invoked
(21.2040)
Cancelled Passengers are enlisted
(21.2050)
Based on pre-defined value option conditions, passengers are sorted and arranged in a preference order
(21.210)
Database on Passenger, Value Options and Airline Operations
All passengers Handled ?
Yes
Stop
No
Get the next passenger
(21.2060)
Get Passenger's preferences and value option bought
(21.2070)
Assess Airline Operations: Costs, Schedule Availability
(21.2080)
Integrate passenger and airline information and optimize across to produce rebooked itineraries that concurrently maximizes value for passengers and airline
(21.2090)
Store the result in the database